

REPORT DOCUMENTATION PAGE

1. Recipient's Reference

2. Originator's Reference

3. Further Reference

4. Security Classification
of Document

AGARD-R-706

ISBN 92-835-0364-3

UNCLASSIFIED

5. Originator

Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title

THE INFLUENCE OF LARGE SCALE COMPUTING
ON AIRCRAFT STRUCTURAL DESIGN

7. Presented at

the 58th Meeting of the Structures and Materials Panel, in Sienna, Italy,
2-6 April 1984.

8. Author(s)/Editor(s)

Various

9. Date

August 1984

10. Author's/Editor's Address

Various

11. Pages

54

12. Distribution Statement

This document is distributed in accordance with AGARD
policies and regulations, which are outlined on the
Outside Back Covers of all AGARD publications.

13. Keywords/Descriptors

Aircraft
Airframes
Structural design

Computation
Computer programs

14. Abstract

This publication comprises three papers which outline how recent advances in large scale computing capacity could affect the process of aeronautical design.

The papers discuss in turn the results of some investigations into the use of vector processing for solving aircraft structural problems, the influence of the new computing systems on computational mechanisms, and the part which can be played in the design process by Artificial Intelligence software. The third paper goes on to discuss how AGARD might respond to the challenge posed by the present situation.

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD REPORT No.706

The Influence of Large Scale Computing on Aircraft Structural Design

NORTH ATLANTIC TREATY ORGANIZATION



DISTRIBUTION AND AVAILABILITY
ON BACK COVER

NORTH ATLANTIC TREATY ORGANIZATION
ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT
(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

AGARD Report No.706
THE INFLUENCE OF LARGE SCALE COMPUTING
ON AIRCRAFT STRUCTURAL DESIGN

Papers presented at the 58th Meeting of the Structures and Materials Panel,
in Sienna, Italy, 2–6 April 1984.

THE MISSION OF AGARD

The mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Exchanging of scientific and technical information;
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Providing scientific and technical advice and assistance to the North Atlantic Military Committee in the field of aerospace research and development;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

The content of this publication has been reproduced
directly from material supplied by AGARD or the authors.

Published August 1984

Copyright © AGARD 1984
All Rights Reserved

ISBN 92-835-0364-3



*Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ*

PREFACE

The Report of AGARD Inter-Panel Working Group WG06 on Large Scale Computing in Aeronautics addresses itself primarily to questions of hardware and makes no recommendations about the provision of software. It is clear that the full benefits of large-scale computing are unlikely to be realised unless the appropriate algorithms are developed and related software is made available.

An Ad-hoc Group of the Structures and Materials Panel has been formed to define specific software aims for SMP related activities and the means by which these might be achieved. To assist in this task, the Ad-hoc Group heard three pilot papers at the 58th SMP Meeting in Sienna, Italy, on 4th April 1984. These papers summarise the present status and outline future prospects on vectorisation for LSC (Thomas), configuration of LSC (Noor) and software for LSC (Morris) and are considered to be of sufficient general interest to warrant publication by AGARD.

C.G.LODGE
Chairman, Ad-hoc Group on
Influence of Large Scale
Computing on Aircraft Structural
Design

CONTENTS

	Page
PREFACE	iii
	Reference
AEROSPATIALE'S EXPERIMENTATIONS OF VECTORIZATION ON SUPERCOMPUTERS FOR AIRCRAFT STRUCTURAL PROBLEMS by J.M.Thomas and J.C.Dunyach	1
IMPACT OF NEW COMPUTING SYSTEMS ON COMPUTATIONAL MECHANICS AND FLIGHT-VEHICLE STRUCTURES TECHNOLOGY by A.K.Noor, O.O.Storaasli and R.E.Fulton	2
LARGE-SCALE COMPUTING IN AERONAUTICAL DESIGN by A.J.Morris	3

AEROSPATIALE'S EXPERIMENTATIONS OF VECTORIZATION ON
SUPERCOMPUTERS FOR AIRCRAFT STRUCTURAL PROBLEMS

by

J.M.Thomas
Head of Structural Methods Department
Aircraft Division

and

J.C.Dunyach
Engineer (Structural Method Group)
Aerospatiale, Dept. des Structures
A/DET/EG/ST
31076 Toulouse Cedex
France

INTRODUCTION

For almost three years, extensive work has been going on at Aerospatiale, Toulouse for the purpose of testing the vectorial and parallel performance of new computer architectures.

Part of this experimentation consisted in adapting existing Structural Stress routines to certain current computers, in particular the CRAY-1 AND CONTROL DATA CYBER 205 vectorial computers. This provided an estimation of the performance throughput gain that can be expected from this type of computer within the context of industrial processing of large programs.

The human and machine cost required for adapting the programs to these computers was also assessed, and a few basic rules applicable to the vectorization of most of the programs were established.

METHODOLOGY

The connection with the above mentioned computers was made at the beginning of 1981 via two networks : one switched network between an alpha-numeric terminal and the CISI CRAY-1 S in PARIS on the one hand, and one satellite connection (CYBERNET network) with the MINNEAPOLIS CYBER 205 (see figure). The disadvantage of this working process was that it required all data files to be typed from the alpha-numeric terminal.

Therefore, it was only used for small test routines producing random data by themselves. For large data file processing, tapes were sent to the various sites involved and handled from the terminal.

After a phase of experimentation on algorithms and simple, essentially mathematical, routines, a vectorization of the ASELF finite element chain used by the AEROSPATIALE Design Office was undertaken.

For practical reasons (lack of FORTRAN compatibility between computers under test and the reference computer i.e. a CDC 750) we concentrated on vectorizing large time-consuming mathematical KERNELS and more particularly on the routine entitled "Stiffness matrix factorization".

VECTORIZATION OF THE "STIFFNESS MATRIX FACTORIZATION ROUTINE" FOR STRUCTURAL STRESS

After discretisation, the finite elements method leads to the computation of a symmetrical system

$$K \cdot D = F$$

Where K is the structural stiffness matrix.

K is therefore symmetrical and of profile form.

By convention, only the upper triangle is stored in the memory, in the most compact possible form. The sub-routine used for solving this system in the case of one or several loading cases is very cp time consuming.

50 % of the total chain running time is frequently spent in this module only. It is therefore essential to vectorize it carefully.

a - Method used

The algorithm used in the ASELF chain is the conventional CHOLEVSKI column algorithm which offers the advantage over iterative methods of retaining the factored matrix.

This saves considerable time when several loading cases are successively applied to the structure (under consideration).

We initially endeavoured to vectorize the routine already developed without trying to alter the algorithm or data format. Owing to its writing, the Cholevsky algorithm can be vectorised easily.

We thus obtained fairly high performance with vectorized versions on both computers (see figure)

b - Analysis of primary results

The Studies involved four structures taken from current Design applications. The specific feature of the first two (ST1 and ST2) was that they fitted into the central memory (main memory plus ECS for ST2) of the reference computer (CONTROL DATA CDC 750)

The gains achieved on these two structures exactly reflect the compared performance of a scalar computer and a vector computer on a lightly vectorised standard program.

It should be noted that these gains are of the order of 10 for the CRAY 1 S and 15 to 20 for the CYBER 205.

Structures 3 and 4 could not fit into any computer memory. However, owing to the differences in main memory sizes of vector computers (1 million 64-bit words) and scalar computers approximately (250 000) the working areas were not comparable.

So the volume of inputs-outputs required for complete routine running was significantly reduced on vector computers.

Furthermore, with the increase in average vector (matrix column) length, both vector computers were closer to their asymptotic performance (this being particularly true for the CYBER 205). The increase in compared performance of the two computers can be thus explained. The total gain reaches 30 for the CRAY-1 and 40 for the CYBER 205.

It can be noted that these results are already very satisfactory and enable the routine running time to be reduced considerably.

c - Optimization of results

To gain a further performance percentage on this module, we had to look deeply inside the two computers to try and make the routine structure match the hardware as much as possible, using all the available vector resources, even by completely changing the data implementation or algorithm structure.

On the CRAY-1, it would have been ideal to write the computation module directly in Assembler language by optimizing the vector aspect ; but the human cost was too high ; so we merely used simple computation sub-routines existing in the CRAY Assembly library. This was done to the prejudice of transportability and required the creation of an equivalent library written in FORTRAN on the front-end computer CDC 750 (to ensure continuous running in the event of a failure of the CRAY. In comparison with the previous version, the gain proved to be relatively small (about 10 % better) but the importance of the sub-routine justified the effort.

As opposed to the CRAY, the CYBER 205 is very dependent on data presentation. We therefore developed and tested factorisation line per line algorithm, transposed from the CHOLEVSKI algorithm which enables triadic operations, to be displayed with high performance on the CYBER 205. Writing changes in sub-routine proved to be rather small and the gain significant (one and a half times that of the previous version on all structures). However, line formatting involves an increase of about 20 % in the volume of data to be processed.

The filling of GAPS is much more involved in case of a line formulation.

A few matrix geometries corresponding to structural types studied in the Design Office, cannot be treated by this method.

Test on other algorithms, in particular conjugate gradient type algorithms, yielded very satisfactory results on both computers (the CRAY in particular), but other factors associated with the nature of routine processing finally led to retaining the CHOLEVSKY algorithm.

OVERALL GAIN ON THE COMPLETE CHAIN

Let's consider the typical case in which the factorization part represents 70% of the program cp time.

Let's assume the remainder of the chain is scalar (hence only taking advantage of the technological gain due to computer cycle time/clock period reduction). Regardless of those pessimistic assumptions, the total gain of the complete chain remains high. In fact, this gain is 6 for the CRAY and 4 for the CYBER 205. The CRAY/CYBER gain discrepancy, which initially appears surprising, is due to the difference in computer cycles (12.5 for the CRAY-1 S versus 20 for the CYBER 205 and 25 for the reference computer).

These extremely pessimistic assumptions do not take account of the reduction in memory access and data transfer times due to the memory extension and the use of new, faster, technologies. It thus appears obvious that such gains will at least be doubled (this will become even more obvious when other time consuming modules will be vectorized)

COMPLETE CHAIN VECTORIZATION

This will be carried out in several steps some of which are already in progress as follows :

- first of all, after the FORTRAN adjustment phase (providing routine compatibility), vectorization of the main modules without altering the algorithms. The remainder of the chain is unaffected. At this stage, the gains should be of the order of 10.

It is possible to anticipate the use of an automatic vectorization routine, known as a Vectorizer, which is capable of reading the Fortran language directly and re-writing certain portions which have been incorrectly vectorized by the compiler (in particular, certain critical inner loops). This solution though not optimum, has nevertheless the advantage of ensuring minimum initial vectorization of the whole code at lower cost and in a reasonable time.

If required, new algorithms can replace insufficiently vectorizable ones.

The solutions to such problems can still be optimized ; certain parts of the calculation chain, such as the assembly (which is one of the most costly mathematical KERNEL) would draw some benefit from being entirely re-arranged and re-formulated. In fact, the analysis as well as the preliminary tests performed on the CRAY show that it is definitely better to process vectors with elements of the same type by doing most of the processing in a vector way, rather than processing each element separately and on a scalar basis.

This necessitates the use of sufficiently large working areas (several hundreds of thousands of words) but any possible resulting gain would be of the order of 25/30. However, the writing of such a module proves to be difficult since the totality of the finite element library must then be reviewed.

- In parallel, data structure adaptation to the computer in accordance with its specific features (virtual memory on the CYBER 205, random files on the CRAY) by taking the vectorial requirements of the main modules into account. This phase will most probably be time consuming. Certain parts of it have already been processed.
- At last, secondary module vectorization without altering the algorithms (this part may be considered as optional when using a vectorizer).

REMARK : INSUFFICIENCY OF VECTORIZATION

Tests have demonstrated that an overall gain exceeding 10 can be expected from an average run of the finite element code (with the most pessimistic assumptions).

This gain added to the increase in available memory area will be sufficient to cover the requirements of the next three years. However, the progress brought about by vector designs faces a problem common to all computers i.e. the impossibility of reducing the computing time of the scalar section when all the vectorizable parts have been processed.

Frequently, a large part of the programs remains, unaltered and cannot be accelerated in vectorial manner. This is particularly true for all the sections that do not contain much computation and for initialization and data transfer sections. In the long run, this scalar KERNEL becomes the most time consuming part of the routine, which is paradoxical and unfortunately irreversible.

To reduce the time needed for running these scalar fragments, we can resort to true parallelism, i. e. : MIMD (Multi Processor Multiple Data) computers made up of a network of processors (possibly vector) sharing several common tasks. Work on national ISIS and MARISIS projects is in progress in France and AEROSPATIALE has already started tests on simulators of these future computers. Although they require an almost complete alteration of the formulation of the current mathematical methods, the results are encouraging and we intend pursuing our efforts in this direction which is very likely that of the future.

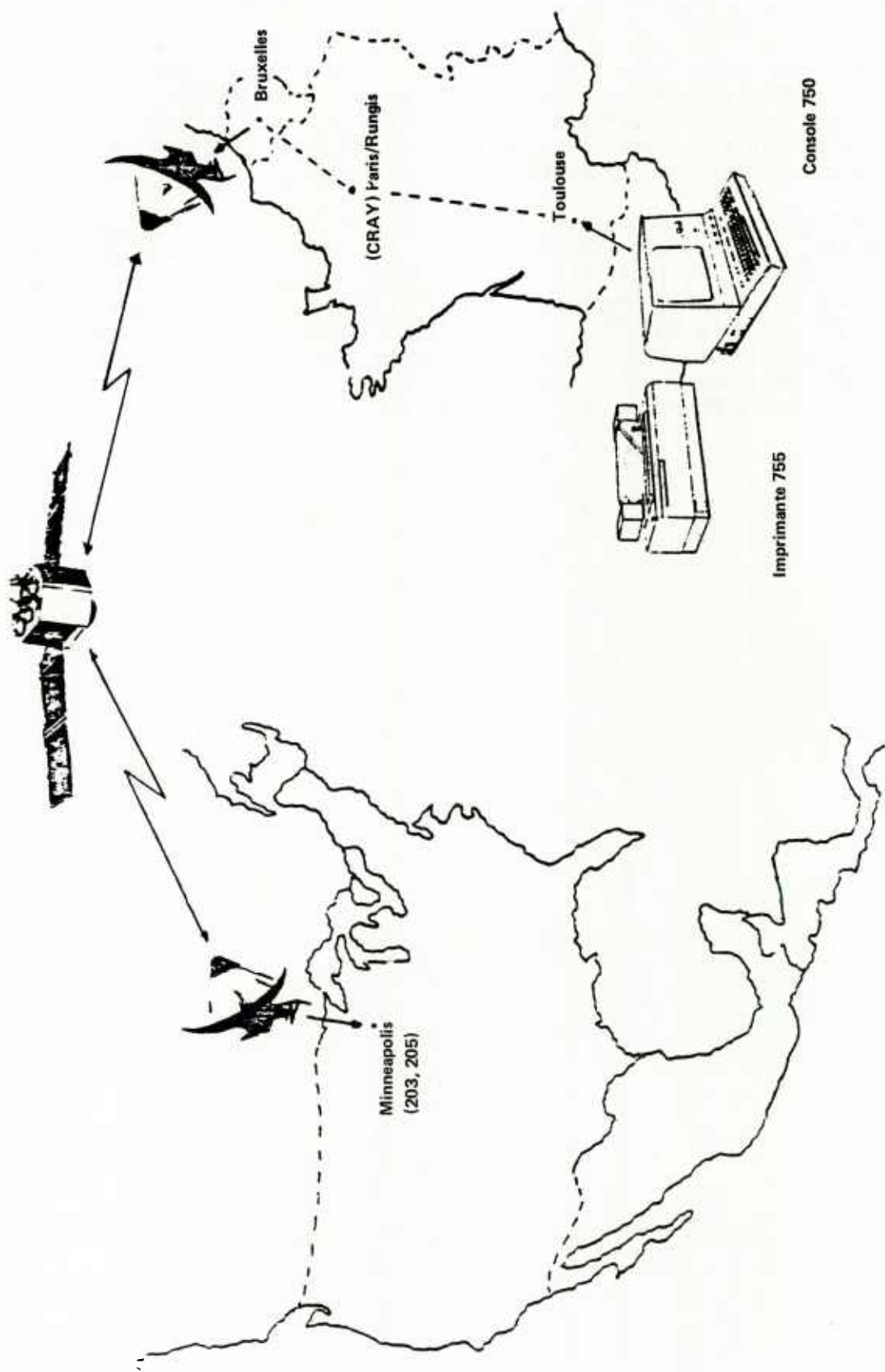


Figure 1

Liaison: Toulouse — réseau commuté Rungis(resseau CYBERNET) — Bruxelles (Satellite) Minneapolis

Figure 2

Structures "Tests" Eléments Finis pour la factorisation

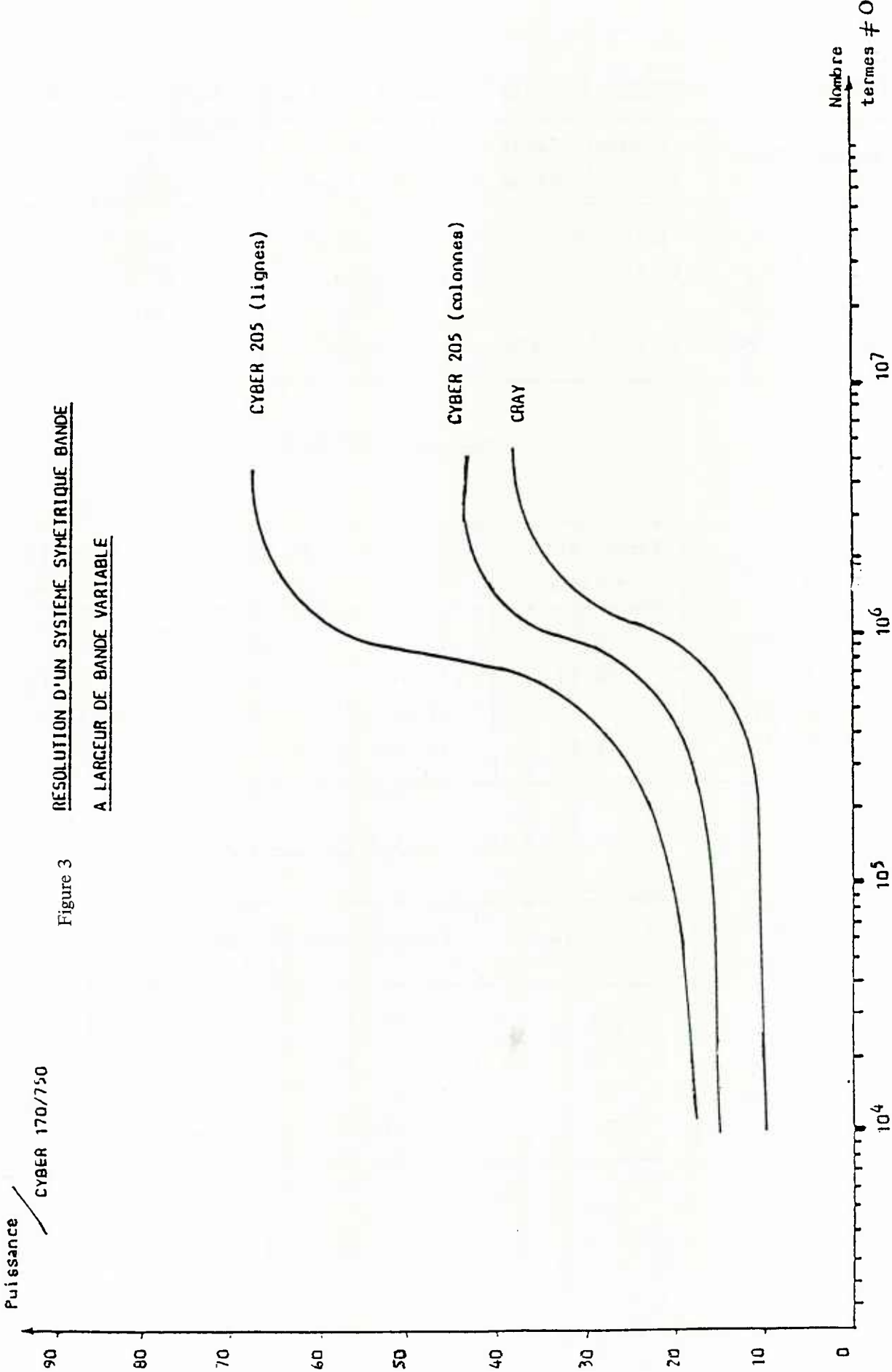
Structure	Ordre	Nb termes ≠ 0	Largeur bande max.	Largeur bande moyenne	Nombre de termes après factorisation	
					colonne	ligne
ST1	277	16350	115	68	19238	19526
ST2	2797	427525	172	158	421812	434247
ST3	2797	1461937	636	584	1371631	1533199
ST4	13580	6926400	1260	520	7184696	8130213

Comparaison CYBER 205/CDC 170-750

Structure	Factorisation colonne	Résolution 1 cas	Factorisation ligne	Résolution 1 cas
ST1	15.26	15	18.02	15
ST2	20.53	18.09	30.44	23.95
ST3	40.07	38.04	63.76	42.04
ST4	44.85	40.05		

Comparaison CRAY/CDC 170-750

Structure	Factorisation	Résolution
ST1	10.42	3.
ST2	11.73	4.5
ST3	30.	10.8
ST4	35.	11.1



ESSAI GIVENS-HOUSEHOLDER 750 ET 205

DEBIT

CALCUL VALEURS PROPRES PAR STURM

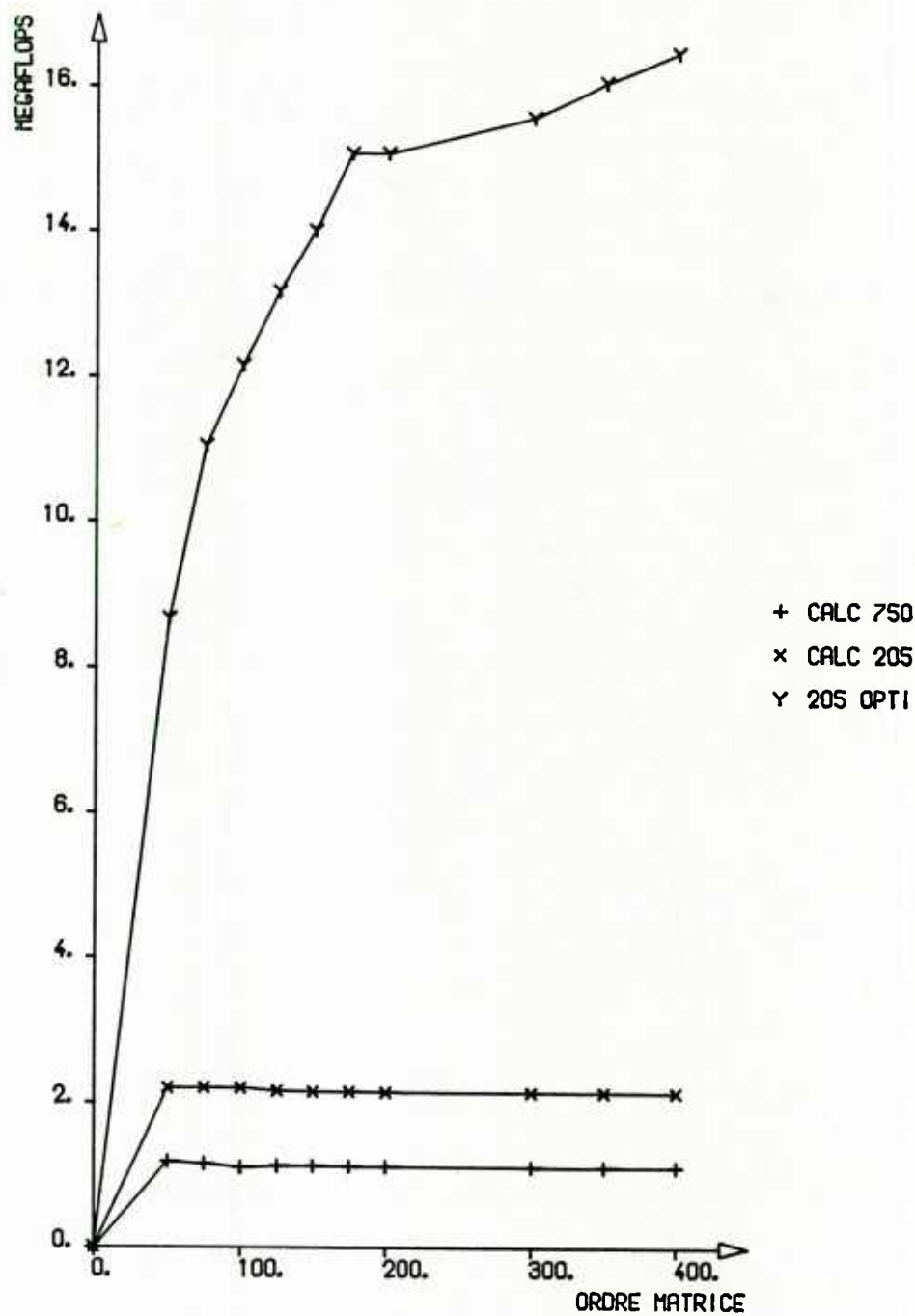


Figure 4

TRIDIAGONALISATION

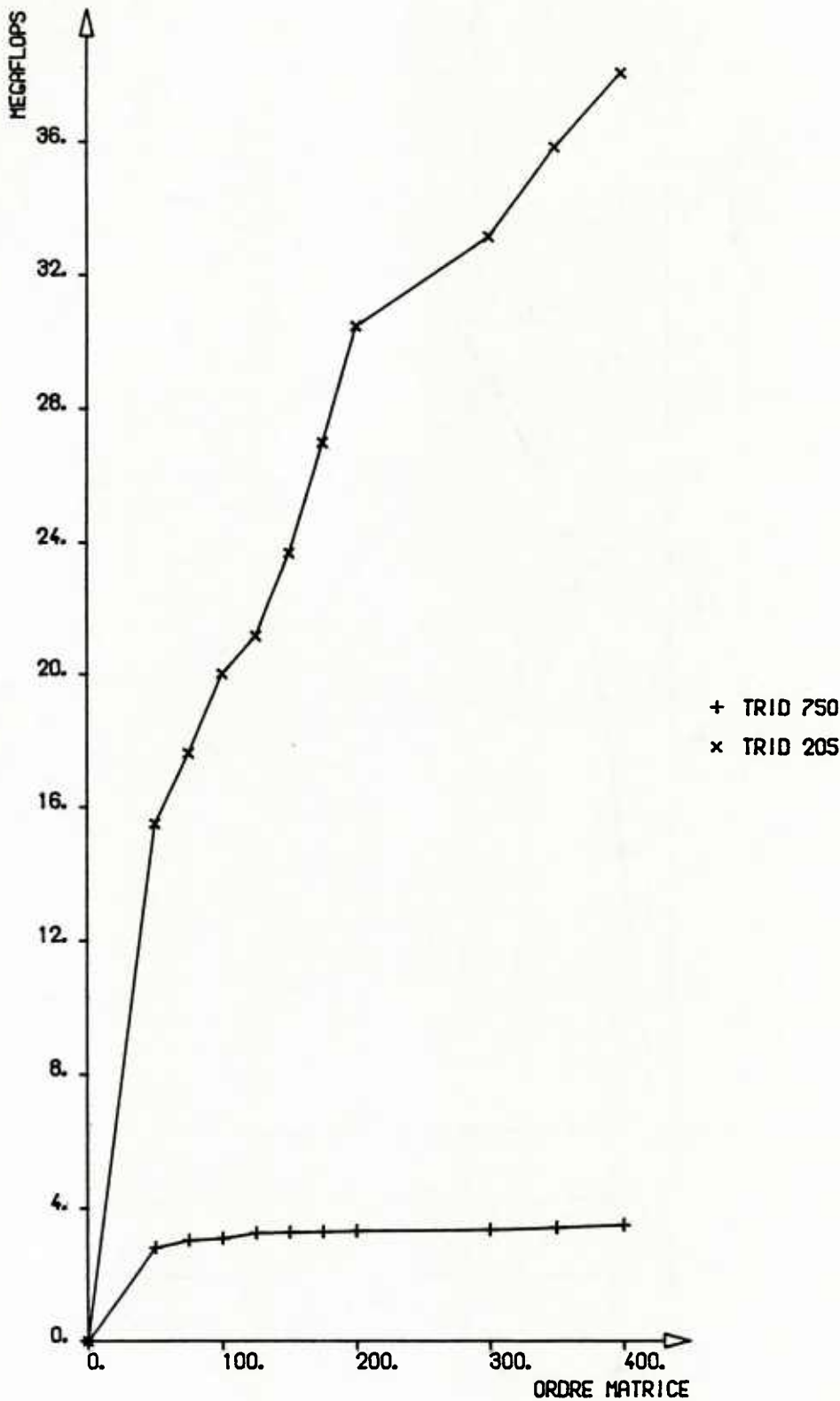


Figure 5

RESOLUTION PAR GAUSS SYM. ET GRADIENT CONJUGUE -TEMPS-
MATRICES BANDES, LARGEUR DE BANDE N/5

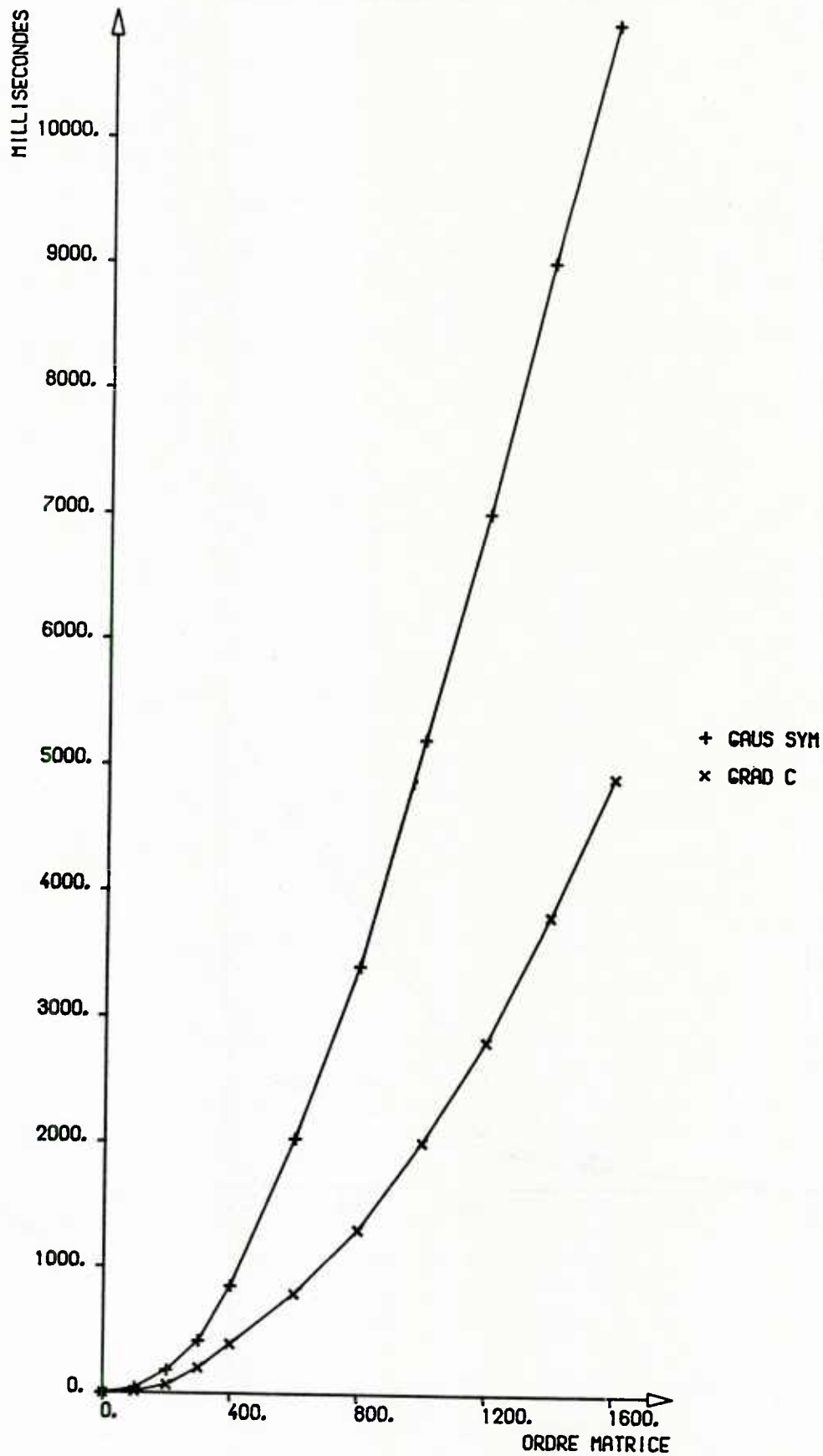


Figure 6

RESOLUTION PAR GAUSS SYM. ET GRADIENT CONJUGUE -TEMPS-
MATRICES BANDES, LARGEUR DE BANDE N/2

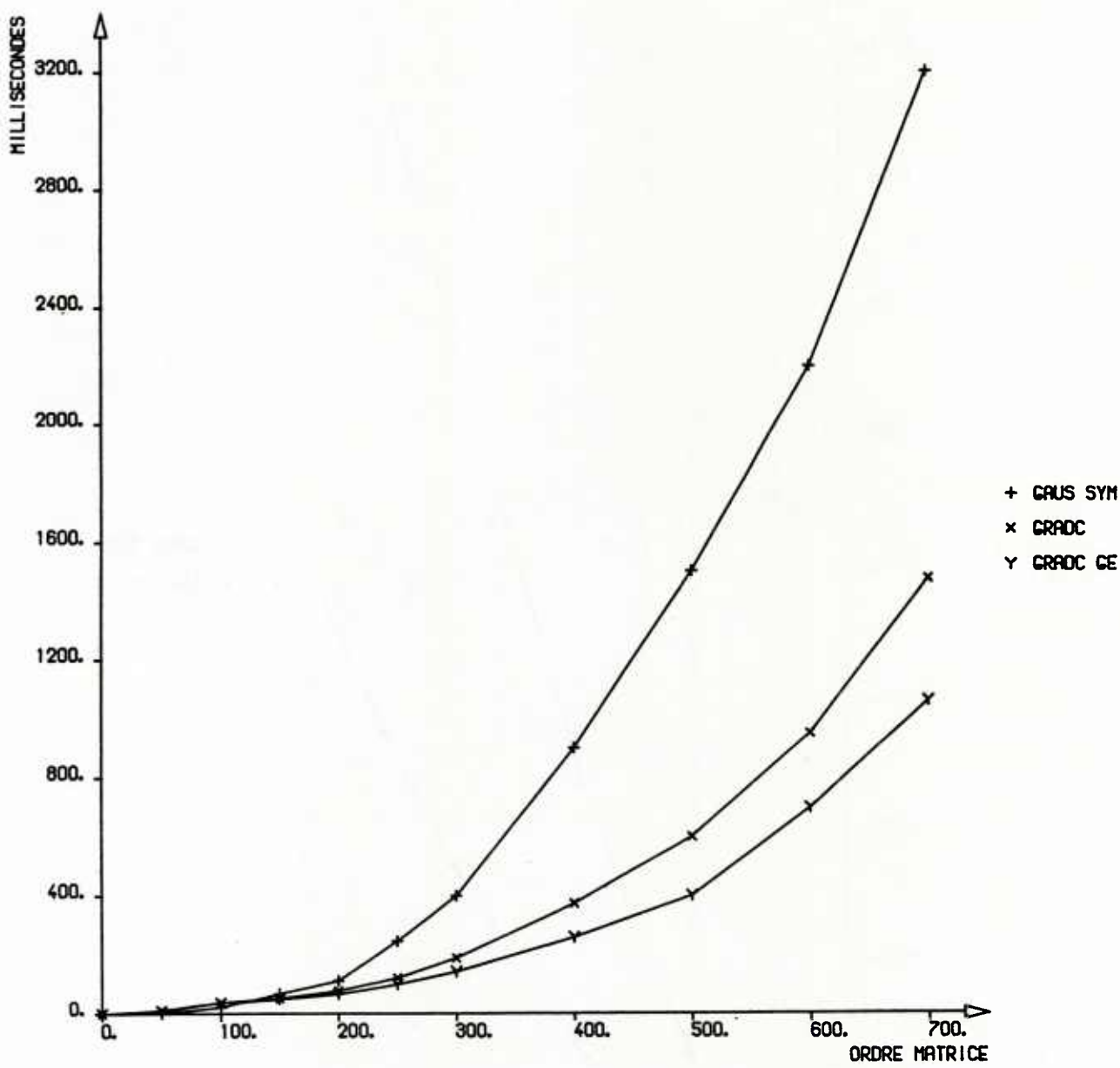


Figure 7

RESOLUTION PAR GAUSS SYM. ET GRADIENT CONJUGUE -TEMPS-
CAS DES MATRICES PLEINES

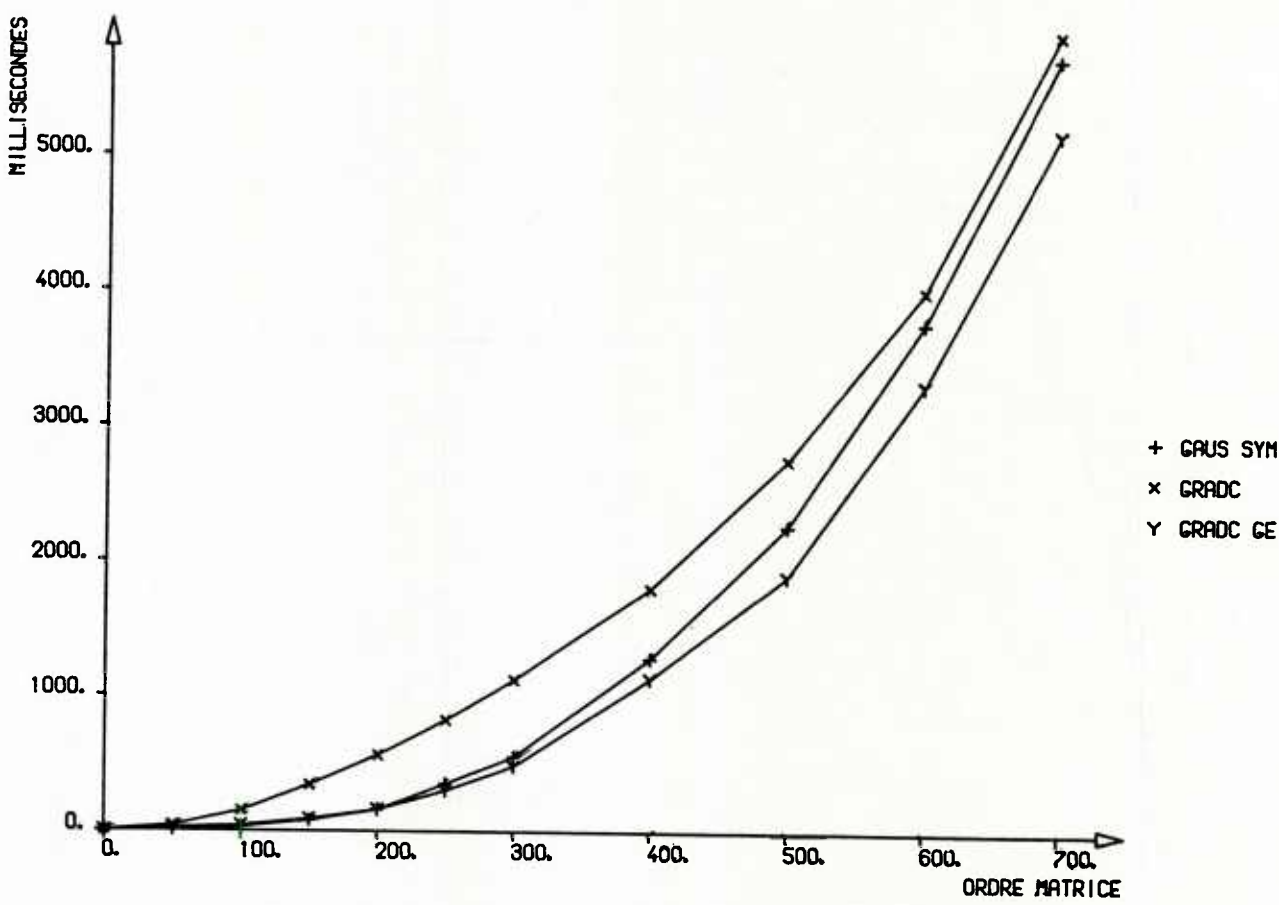


Figure 8

GAUSS SUR MATRICE PLEINE -DEBIT-
QUATRE METHODES DE PROGRAMMATION

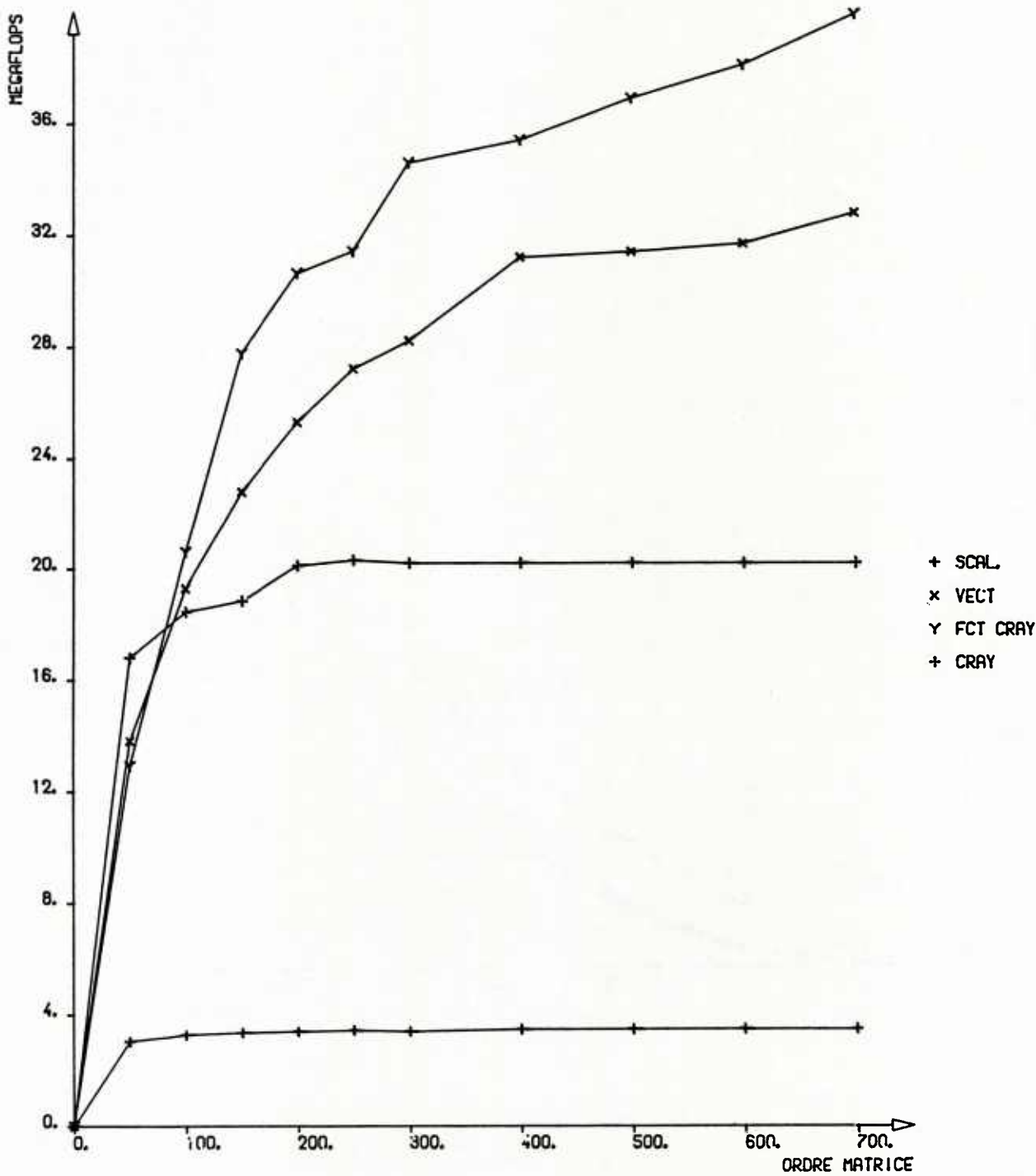


Figure 9

IMPACT OF NEW COMPUTING SYSTEMS ON COMPUTATIONAL MECHANICS AND FLIGHT-VEHICLE STRUCTURES TECHNOLOGY

Ahmed K. Noor, Olaf O. Storaasli and Robert E. Fulton
Mail Stop 246
NASA Langley Research Center
Hampton, Virginia 23665
U.S.A.

ABSTRACT

Recent advances in computer technology that are likely to impact computational mechanics and flight-vehicle structures technology are reviewed. The characteristics of supersystems, highly parallel systems, and small systems (mini, microcomputers and engineering workstations) are summarized. The interrelations of numerical algorithms and software with parallel architectures are discussed. A scenario is presented for future hardware/software environment and engineering analysis systems. A number of research areas which have high potential for improving the effectiveness of analysis methods in the new environment are identified.

1. INTRODUCTION

The last two decades have witnessed an explosive growth in computer technology. This growth shows no sign of abating; all the indications are that the changes during the next decades will prove to be even greater, particularly with the introduction of novel forms of machine architecture (e.g., vector, array and multiprocessor systems). The computer hardware developments are most noticeable at the two extremes of the spectrum. At one end, the large expensive computer systems, usually referred to as supersystems, such as CRAY 1S, CDC CYBER 205 and Denelcor HEP1 have radically different and novel architectures resulting in very high performance (computational speed of the order of 100 million floating point operations per second (100 MFLOPS) or more). At the other end of the spectrum are the various types of minicomputers, engineering workstations, microcomputers, handheld computers, and microprocessors.

The introduction of these new computing systems has made a strong impact on computational mechanics. The large supersystems have made possible new levels of sophistication in the modeling of flight-vehicle structures as well as in problem depth and scope which were not possible before. More importantly, however, the new supersystems have opened up a view of the next steps to be taken in the development of engineering analysis software/hardware systems. In order to realize the full potential of the supersystems in the analysis of flight-vehicle structures, special parallel numerical algorithms, programming strategies and programming languages need to be developed.

The small, low-cost computer systems (mini, microcomputers and engineering workstations) provided a high degree of interactivity and freed the analysts from the constraints that are often imposed on them by large centralized computation centers. However, the effectiveness of the small computers for solving large-scale flight-vehicle structures problems is limited. The combination of a fast attached processor with a mini (or micro) computer into a dual-processor system can extend the range of flight-vehicle structures problems that can effectively be solved by the minicomputer. In recent years several general-purpose analysis programs, notably finite element programs, have been made operational on the minicomputers and new finite element software is currently being developed for microcomputers.

The objectives of the present chapter are: a) to discuss key engineering analysis requirements placed on evolving computer technology; b) to summarize some of the developments in computing systems during recent past and near-term future and relate these developments to computational mechanics and flight-vehicle structures technology; and c) to outline the likely directions of engineering analysis software/hardware systems over the next five to ten years.

A number of previous attempts have been made to predict the characteristics of future engineering analysis systems, notably finite element systems, and the impact of the advances in computing systems on finite element analysis (see, for example, Refs. 1 to 7). The discussion presented herein is much more detailed than that given in the cited references because the authors feel that engineers need to become familiar with the characteristics of the new computing systems in order to realize their full potential in advanced analysis methods in general, and in flight-vehicle structures computations in particular.

2. TECHNICAL NEEDS FOR ENGINEERING COMPUTATIONS

The driving force for future developments in engineering computations will continue to be the need for improved productivity and cost-effective engineering systems. To achieve this goal, the following sets of technical needs can be identified:

1) Expanding the scope of engineering problems considered. This includes examination of more complex physical phenomena (e.g., progressive failure of fibrous composite structures); study of interaction phenomena (as would be required in the hydrodynamic/structural coupling in deep sea mining and the thermal/control/structural coupling in space exploration); and the prediction of the reliability of the model's performance to uncertainties in its input parameters. Such reliability-based analysis can help in making rational design decisions and in achieving realistic cost/performance trade-offs.

2) Development of a hierarchy of models, algorithms and procedures for engineering systems. Simplified and specialized models and algorithms are appropriate for use in the preliminary and conceptual design phases and more sophisticated models are used in the detailed design phase.

3) Development of effective integrated design, synthesis and evaluation tools to support all phases of product development including the design, management and manufacturing cycle.

4) Continued reduction of cost and/or time for obtaining solutions to engineering design/analysis problems.

The hardware and software requirements to meet the aforementioned technical needs include:

a) A spectrum of available computer capabilities ranging from high-performance computing systems for large complex problems to low-cost engineering workstations which provide convenient computer capabilities at the engineer's desk.

b) Distributed asynchronous processors which carry out several different calculations in parallel using the same or different code and referencing the same or different databases.

c) User-friendly hardware interface or engineering workstation with the following capabilities: high resolution and high-speed graphics; high-speed long distance communication; and verbal (audio) as well as visual interfaces.

d) Artificial intelligence-based expert systems, incorporating the experience and expertise of practitioners, to aid in the modeling of the engineering system, the adaptive refinement of the model, and the selection of the appropriate algorithm and procedure used in the solution.

e) Powerful engineering database management system.

f) Turnkey engineering application software systems which have advanced modeling and analysis capabilities and are easy to learn and use.

The implication of these requirements on computational mechanics along with a scenario of future engineering analysis systems are discussed in another section. An assessment of future computer system needs for large-scale computation in a number of technical areas has been summarized in Ref. 8.

3. BRIEF REVIEW OF CURRENT AND PROJECTED ADVANCES IN COMPUTER TECHNOLOGY

The major computer technology developments have been, and continue to be, focused on improvements of cost, size, power consumption, speed and reliability of electrical components. These developments can be classified into two general areas: the hardware components and the computer architecture and system design methods (Ref. 9). The most notable advances in hardware components in the last two decades have occurred as a result of developments in microelectronics. Instead of connecting discrete components together by wires to produce a circuit, complete circuit patterns, components and interconnections, are placed on a small chip of semiconductor material (usually silicon); see Fig. 1. The principal advantages of microelectronic circuits are their reliability, low cost and low power consumption. The trend of ever-increasing the number of devices packaged on a chip has given rise of acronyms SSI, MSI, LSI, VLSI and ULSI standing for small-scale, medium-scale, large-scale, very large-scale and ultra-large scale integration, respectively. The beginning time frame for each of these technologies along with the chip density ranges are listed in Table 1. Beginning in 1960, the number of components on a chip has increased, more or less, exponentially with time. For the case when no differentiation is made between logic and memory the progression is shown in Fig. 2.

The full range of hardware components (computer building blocks) are now available on microelectronic chips; these include memory units, addressing units (i.e., counters and decoders) and complete central processing units (CPU) called microprocessors; and even complete microcomputers (which include the CPU, memory, and input/output functions all residing on a single chip). Significant architectural advances resulted from imaginative ways of configuring these basic blocks.

The net effect of the aforementioned developments has been a significant decrease in the cost of performing computations. This is illustrated in Fig. 3 where it can be seen that the cost of performing a given calculation has decreased by a factor of ten every eight years (Ref. 10).

In the present and succeeding sections the major advances in hardware components are reviewed and some of the new computing systems are briefly described. The survey given here is by no means complete or exhaustive; the intention is to concentrate primarily on those developments which have had, or promise to have, the greatest impact on the manner in which flight-vehicle structures components are performed. Discussion is focused on semiconductor technology; processor speed; memory organization, secondary storage devices; access facilities and user interface; and networking.

3.1 Semiconductor Technology

The predominant semiconductor material in use to date is silicon. Better understanding of this material as well as better processing, tooling and packaging techniques enabled the design of fast dense circuitry. Current semiconductor technologies fall into two categories (Ref. 11):

Logic made from bipolar transistors which are current controlled. Three classes of Logic can be identified in this category; namely: Transistor-Transistor Logic (TTL); Emitter Coupled Logic (ECL); and Integrated Injection Logic (I²L). The first is used in SSI and MSI. The second is suitable for LSI and the third is good for VLSI.

Logic made from field effect transistors (FET) which are voltage controlled. A number of classes can be distinguished within this group including metal-oxide semiconductor (MOS); complementary MOS usually referred to as CMOS; and MOS devices fabricated in silicon grown on sapphire SOS/MOS. The logic circuit integration level attainable by each of the aforementioned technologies is depicted in Fig. 4.

Current research is directed towards: a) shrinking of conductor and device dimensions (scaling) to micron and submicron dimensions (Ref. 12); and b) increasing the speed of logic circuits (to achieve a machine cycle time of the order of one nanosecond (1×10^{-9} second) - Ref. 13).

The first objective can be accomplished by using recent and improved lithography tools (including optical, electron beams, UV optics, direct-write electron beam, X-ray, and ion beam techniques). Three candidate technologies are likely to achieve the second objective of ultra-fast logic circuits. These technologies are: a) FET technology using gallium arsenide (GaAs), a component semiconductor material instead of silicon; b) three-dimensional integrated circuit design; and c) Josephson Junction devices which consist of sandwiches of insulating oxides between two strips of superconducting metals. Heat dissipation is still an important limitation with gallium arsenide. Josephson Junction devices with several orders of magnitude less power dissipation, essentially remove this limitation. However, their drawback is the intense cryostatic environment (cooling to 4°K above absolute zero) required for the operation of these devices.

3.2 Memory

Memory is the most rapidly advancing technology in microelectronics. Recent progress includes development of an entire hierarchy of addressable memories, and of high-speed, random access memory chips with many bits of data. Each level in the hierarchy represents an order of magnitude decrease in access speed, and several orders of magnitude increase in capacity, for the same cost. The techniques of splitting and interleaving among various types of memory hierarchies in individual systems have changed some of the basic concepts of computing itself. Instead of just a few registers in the central processing unit (CPU) and a single-level memory, a typical machine may now have:

- a) A number of high-speed, general-purpose registers;
- b) A cache memory for very rapid access to small amounts of data or instructions;
- c) Standard central or equivalent memory;
- d) Extended memory, directly addressable, but at a lower speed;
- e) Hardware-implemented virtual memory, extending the amount of addressable space.

Initially, memories were based on bipolar transistor technology. However, by 1980 MOS technology accounted for over 80 percent of the semiconductor memory market. CMOS devices are becoming increasingly important because of their lower power requirements.

There are several types of semiconductor memories. These include Random Access and Read Only memories. In Random Access Memory (RAM), data can be written into or read out of any storage location in random fashion without regard to its physical location relative to other storage locations. Read Only Memory (ROM), contains a permanent data pattern stored during the manufacture of the semiconductor chip in the form of transistors at each storage location that are either operable or inoperable. RAM can be either static or dynamic. Dynamic RAM (DRAM) requires constant refreshing to maintain its data, which static RAM (SRAM) does not. However, advances in DRAM permit double the amount of RAM at about the same cost as static RAM, so DRAM is used more frequently.

Figure 2 shows the projected increase in the density of dynamic RAM chips from their current value of 256K bits (256,000 bits) to 4MB (4×10^6 bits) in 1990 on roughly a linear scale. These improvements take place by factors of four merely by reducing the linear dimensions in half in each direction which results in one quarter the area with an added benefit of reduced power requirements.

The increase in the chip density of RAM, from their current value of 256K bits to their projected value of 4MB in 1990, can be exploited by either reducing the number of chips on the memory board, or more likely, by increasing the user-memory capacity. The increase in the memory capacity is depicted in Fig. 5 for handheld computers with eight-chip memory board and desktop computers with 128-chip memory board. For the sake of comparison, the memory capacity of the current and planned supersystems are also shown in Fig. 5. The data presented in Fig. 5 is based on projections made in the current electronics literature. The impact of increased memory for flight-vehicle structures analysis in the future is quite significant since memory capacity directly translates to increased problem size capability. Developments with low power complementary metal oxide semiconductor (CMOS) memory and higher speed obtained by using gallium arsenide instead of silicon, promise additional benefits for specialized applications. However, for most applications, dynamic RAM in high densities is expected to be of major benefit to engineering analysts since it will permit larger problems to be solved at reduced cost.

3.3 Microprocessors and Chip Technology

The advances in microprocessor miniaturization currently allow several dozen layers (like floors in miniature buildings) in a single microprocessor CPU chip with hundreds of thousands of equivalent transistors on each layer. In addition to more compact CPU chips, the speed at which they are capable of operating will continue to increase. This is important to flight-vehicle structures computations since the clock speed at which microprocessors can operate is directly proportional to the number of floating point operations possible per second (the speed of computation). Early microprocessors had a clock speed of 1 MHz, which increased to 12 MHz in current microprocessors and, based on current projections in the electronics literature, future speeds of over 50 MHz are expected (Fig. 6). The increase in the clock speed, measured in MHz, directly affects the throughput of engineering calculations measured in terms of the number of floating point operations per second (FLOPS). The use of new materials such as gallium arsenide circuits in CPU chips is also expected to increase the CPU speed significantly. The impact of VLSI is likely to continue to play the most dominant role for immediate application in new products, while research on Josephson Junction devices requiring "supercool" environments may find application to some special purpose computers where floating-point operations in the nanosecond range may be achievable.

Two major research efforts are currently underway for improving the processor speed (see Refs. 14 and 15). The two projects are labeled VHSIC (very high-speed integrated circuits) and VHPIC (very high-performance integrated circuits). The first is sponsored by the U.S. Department of Defense and the latter is supported by the British.

The new powerful chips resulting from these efforts can be used as onboard monitoring systems for flight-vehicle structures to augment stability, reduce loads, improve ride comfort, and minimize aerodynamic drag, thereby allowing more economical designs. The new microprocessors can also be used as monitoring systems for the detection, recording and evaluation of stochastic damage. These monitoring systems can be instrumental in supporting service requirements and increasing the mean time between inspections.

3.4 New Hardware Architectures

In addition to increasing the speed of single processors in a control-flow architecture (conventional computers), two design options for faster processing exist. The two options are based on:

- a) Using a few very fast processors and enhancing the control-flow architecture with pipelines (operating in an assembly-line manner) and some specialized processors; and,
- b) Using a large number of fast or medium-speed off-the-shelf processors to form highly-parallel systems.

Examples of computing systems based on these two design options will be given in the succeeding section.

Many of the new computers (from supercomputers to micros) contain more than one processor for improved performance. This trend is likely to continue as processors become easier to design and more compact. Before microprocessor CPUs were invented the most laborious part of developing new computers was the signal backplane and digital component wiring of a multiboard CPU. Now that the CPU is on a VLSI chip and associated chips are available for memory management, input/output and other functions formerly taking up one or more boards, the entire CPU and associated functions are being condensed to one board with additional boards for memory. In the case of microcomputers, all functions including memory are reduced to a single board. It is expected that the majority of computers in the future will be single-board computers and if additional boards are required they will be for memory or peripheral controller functions only.

3.5 Secondary Storage Devices

Significant advances are taking place in secondary storage technology. The advances reported in the literature for disk technology are depicted in Fig. 7. While the conventional hard disks (Winchester disks) are increasing in density of storage (currently in excess of 1 Gigabyte (1×10^9 bytes)) and floppy disks are approaching 1 MByte, the optical disk promises to make the most dramatic impact on computer secondary storage. For example, the capability to read and write rapidly and inexpensively 20 Gigabytes (i.e., 20 billion characters) of data (equivalent to the contents of the Encyclopedia Britannica) promises to provide an attractive and cost-effective storage for large volumes of data in relatively long-term project files typically used for multidisciplinary design projects. The future architectures promise to offer specialized hardware (e.g., chips) for fast floating-point calculations, interactive graphics, algebraic manipulation, input/output processing, and most other standard procedures, eventually including finite element analysis and matrix operations on silicon or gallium arsenide chips.

3.6 User-Interface Hardware and Software

A great deal of effort is currently taking place to improve the productivity of the analyst by developing user-friendly software and hardware interfaces. More finite element software systems are becoming "turnkey" with defaults built-in, and with simple menu options. Future menu options are likely to be multi-window (one window for each task) and be controlled by lightpen, voice or mouse (which is an advanced user-friendly capability for accessing the system). The finite element model can be generated by using either one of the geometric modeling software packages or a CAD system. The model data can be changed whenever desired via voice or mouse.

3.7 Distributed Computing and Networking

The first large-scale computer network, the ARPANET, was initiated in the mid-sixties. The major objective of this network was to explore the possibilities and implications of large-scale distributed computing, where the analyst at one location could have direct access to several computing facilities and/or databases at a variety of remote locations. This initial work has pointed the way to the feasibility and practicality of distributed computing. However, a tremendous amount of work remains to be done in developing uniform protocols, standard control structures and uniform data description standards before distributed computing can be done routinely on a large scale. When these problems are solved, as they undoubtedly will be, many applications and extensions of distributed computing will take place. For example, distributed computing can be recursively applied to component subtasks according to a number of criteria.

Two recent developments will greatly enhance distributed computing. The first is the widescale availability of microcomputers and engineering workstations, and the second is the new communication technology based on packet switching, which has already emerged. In this technology information is transmitted in packets, typically a few hundred bytes. This is the first communication technology geared to data transmission and is likely to permit economic implementation of distributed computing. The coupling of digital networking with the existing telephone and PBX exchange systems into integrated-services digital networks (ISDNs) promises to offer access to a wide range of data and central computers via desktop workstations.

4. MAJOR FEATURES OF NEW COMPUTING SYSTEMS

Because of the rapid progress made in recent years in semiconductor technology, a number of novel forms of computer architectures have emerged. In order to put the new computer architectures in proper perspective, a classification of computing systems is needed. Several classifications have been proposed in the literature based on the mode of processing instructions and data; and the mode of interaction between different processors (see, for example, Ref. 11). One of the earliest and most commonly-used classifications is that introduced by Flynn (Ref. 16), and is based on how the machine relates its instructions to the data being processed. A stream is defined as a sequence of items (instructions or data) as executed or operated on by a processor.

Four broad classes can be identified according to whether the instruction or data streams are single or multiple (Fig. 8):

- a) Single instruction stream/single data stream (SISD) machines. These include the conventional serial computers which execute instructions sequentially, one at a time.
- b) Single instruction stream/multiple data stream (SIMD) machines. These are computers that have a single control unit, a collection of identical processors (or processing elements), a memory or memories, and an interconnection network which allows processors to exchange data. An example of an SIMD machine is depicted in Fig. 9. During execution of a program the central control unit (CU) fetches and decodes the instructions and then broadcasts control to the processing elements. Each processor

performs the same instruction sequences, but uses different data. These operations are usually referred to as *lockstep operations*.

c) Multiple instruction stream/single data stream (MISD) machines. There is some disagreement on the computers belonging to this classification. The various pipeline (or vector) processors, which segment computations into consecutive stations, are possible candidates of this class (Ref. 18). However, in many computer science publications pipeline processors are classified as SIMD machines. In pipeline processors, the basic arithmetic operations are broken up into a set of elementary steps which, when performed in series, achieve the desired operation. Each elementary step is then implemented into hardware and the resulting arithmetic unit operates as a production (or assembly) line called the pipeline.

d) Multiple instruction stream/multiple data stream (MIMD) machines. These are computers which contain a number of *interconnected processors*, each of which is programmable and can execute its own instructions. The instructions for each processor can be the same or different. The processors operate on a shared memory (or memories), generally in an asynchronous manner. The interconnection between processors distinguishes an MIMD machine from a set of independent computers (see Fig. 9). The computers of the MIMD class are the most general of all new computers and include all forms of multiprocessor configurations, from linked mainframe computers to large arrays of microprocessors. However, the increased flexibility is at the cost of increased synchronization, overhead, and programming complexity.

Examples of computing systems which belong to each of the four classes are given in Fig. 8.

In this section, the major features of some of the new computing systems which can have a strong impact on flight-vehicle structures computations are reviewed. For convenience, the computing systems considered are divided into five groups:

- a) supersystems;
- b) highly-parallel systems;
- c) special-purpose computing hardware;
- d) small systems; and,
- e) attached processors.

The first group consists of the general-purpose machines with very-high throughput performance. The machines of the second group achieve high performance through extensive use of LSI, VLSI and ULSI memory chip technology. The special-purpose computing hardware considered herein is a subclass of the highly parallel systems and is discussed separately because of its high potential for cost-effective, large-scale flight-vehicle structures computations. The small systems considered include the various minicomputers, engineering workstations, microcomputers (e.g., desktop and handheld personal computers). The attached processors are fast backend computers which, when combined with the minicomputers (or the micros), can extend the range of the finite element problems that are solved effectively on these small systems. Some of the computing systems considered herein are commercially available, others are still research tools aimed at achieving high performance and/or low-cost computations.

4.1 Supersystems

Supersystems are a class of general-purpose computers designed for extremely high throughput performance. The three major characteristics of supersystems are (see Ref. 19):

- a) High computational speeds (maximum speeds of the order of 50 MFLOPS or more);
- b) Large main (or central) memory (with a capacity of 8M Bytes or more); and
- c) Fast and large secondary memory with a sophisticated memory management system.

The development of supersystems now spans two generations. The first generation included the array of processors ILLIAC IV (SIMD machine); and the pipeline (or vector) computers CDC STAR-100 and Texas Instruments Advanced Scientific Computer (ASC). The second generation supersystems used a hybrid combination of pipeline and array processors to achieve peak computational speeds in excess of 100 MFLOPS. Examples of these supersystems are the CRAY-1S; the CRAY X-MP; the CDC CYBER 205; and the Denelcor Heterogeneous Element Processor (HEP1). The latter is the only general-purpose commercial MIMD machine built. As of March 1984, there are 60 CRAY-1, 8 CRAY X-MP, 24 CYBER 205 and 7 Denelcor HEP1 installed. The top sustained speed of the fastest machines today is in the neighborhood of 25 MFLOPS, with bursts to 160 MFLOPS. The next generation of supersystems currently under design include CRAY 2 (successor to CRAY 1 with planned delivery in 1985); GF-10 of ETA (formerly CDC CYBER 2XX) (1986); CDC Multiparallel processors CYBERPLUS; Denelcor HEP 2 (1986); the Japanese supersystems: Hitachi S-810/20 (1984) and FUJITSU FACOM VP-200 (1985); the Japanese Super Speed Computer Project aimed at developing a machine with peak performance of 10 GigaFLOPS (1989); the Japanese fifth generation project

which appears to be aimed at establishing an artificial intelligence industry by the mid 1990's (Ref. 20); and the NASA Numerical Aerodynamic Simulation capability (NAS). The latter is designed to perform most of the calculations required to design a new aircraft, thereby eliminating the need for wind tunnel and flight testing. It is estimated that by 1984 a NAS capability will be built with a sustained speed of 250 MFLOPS and a high-speed memory of 64 M. words. The NAS capability is planned to be upgraded in 1987 to a sustained speed of 1 GigaFLOP and a memory of 256 M. words. Some of the major characteristics of the current and projected CRAY, CDC CYBER, and Denelcor HEPL supersystems are summarized in Table 2. Also, Fig. 10 shows the trend in the computational speeds provided by the United States and Japanese supercomputers. Indications are that computational speeds will continue to increase, and will exceed 20 GigaFLOPS before the end of the present decade. Speeds in excess of 1 GigaFLOP are expected to be achieved by MIMD architectures.

In spite of a number of successful finite element applications on the first-generation supersystems, these supersystems did not live up to their expectations in large-scale finite element computations. This is because the ILLIAC IV operates efficiently *only* when performing an identical operation over an entire array, and the STAR-100 operates efficiently *only on long vectors*.

On the other hand, the second generation supersystems achieve reasonable performance with scalar and short-vector computations. Nonetheless, the efficient use of these supersystems in large-scale finite element calculations requires new ways of structuring the finite element procedure, new algorithms, and new software tools. Direct conversion of finite element programs from sequential computers does not realize the full potential of these supersystems (Ref. 21). A number of large-scale finite element programs have already been installed on the CRAY-1 computer. These include ANSYS, ASAS, ASKA, EISI-EAL, MARC, MSC/NASTRAN, PAFEC 75, and STAGS.

4.2 Highly Parallel Systems

The insatiable demand for increased computer performance, over what can be provided by circuit speed alone, coupled with the advances made in the design and fabrication of VLSI circuits have led to the introduction of parallelism in a number of new computing systems. Parallelism refers to the ability to overlap or perform simultaneously many of the tasks performed by the computer (e.g., memory fetch or store, arithmetic or logical operation and I/O operation). There are three principal ways of introducing parallelism into the architecture of computers, namely (Ref. 11):

Pipelining. Based on the application of assembly-line techniques to improve the performance of an arithmetic or control unit.

Functional Replication. Through providing several independent units or performing different operations such as logic, addition or multiplication, and allowing these units to operate simultaneously on different data.

Array Replication. Through the use of several processors to perform the computations. The processors can form either an SIMD or MIMD machine. In the first case identical processors are used under common control, all performing the same operation simultaneously on different data. In the second case several asynchronous processors are used, each obeying its own instructions, with some mechanisms for interprocessor communication, and local and global control.

Highly-parallel systems composed of a large number of computing elements can be classified into the following groups (Ref. 22):

- a) Multiple special-purpose functional units;
- b) Associative processors;
- c) Array of processors;
- d) Multiprocessors (with multiple CPUs); and,
- e) Data flow processors.

For their specific tasks, multiple special-purpose functional units are the fastest, but they are the least general. They can be designed for almost 100 percent hardware efficiency (in the sense that a large percentage of the hardware is in use at any given time) and require little or no software to perform their unique tasks. In comparison with the multiple special-purpose functional units, the other machines are more general. However, the more general the machine is, the lower the hardware efficiency (in the sense described above), and the more complicated the software needed is likely to get.

Examples of the aforementioned machine types are given subsequently. Note that to date there is no truly general-purpose parallel architecture to which all algorithms can be effectively mapped.

4.2.1 Multiple Special-Purpose Functional Units

These are useful for performing computationally intensive basic mathematical operations such as matrix multiplication and solution of linear algebraic equations. An example of these machines is provided by *systolic arrays* in which high-level computations are directly mapped into hardware structures. Systolic arrays are based on the principle of replacing a single processing element (cell) by an array of processing elements (cells), in order to achieve a high computational throughput without having to increase the memory bandwidth (Ref. 23). This is depicted in Fig. 11.

In a systolic system data flows from the computer memory in a regular fashion, passing through many processing elements with built-in hardware instructions before it returns to memory. Currently available systolic arrays have four basic characteristics, namely:

- a) They involve only few types of simple cells.
- b) Data and control flows are simple and regular.
- c) Algorithms used support high degrees of pipelining and parallelism.
- d) Each input data item is used in a multiplicative and effective manner.

A list of prototype realizations of systolic arrays for various algorithms is given in Ref. 24. Potential applications of systolic arrays for finite element computations are discussed in Refs. 25 and 26.

4.2.2 Associative Processors

In these processors parallelism is introduced in the memory organization by using the bit-slice scheme. This is depicted in Fig. 12 for a w word memory with word width b . In conventional computers, access is restricted to sets of b bits which form a word, no other set of b bits can usually be accessed in parallel. On the other hand, in the bit-slice parallel computers, the memory slice accessed consists of s bits, with only one bit from each word (Ref. 11). This structure is orthogonal to that of the conventional machines, or would be if s were equal to w . In general, s will be very large, measured in thousands. Since one bit of any memory word is available on one access, it is possible to search the whole memory simultaneously for specified contents by iteration on bit slices. This processor-memory organization is used to select memory words by their contents, or by some attribute of a subfield of their contents rather than by address. The parallel by-bit-slice memory is usually referred to as an associative memory.

Many machines have been designed around the concept of an associative memory, except that in addition to data searching circuitry, each memory word or group of words has a simple arithmetic unit. This element can perform numerical comparisons as well as arithmetic operations. An example of associative machines is provided by the Goodyear Aerospace Staran, which is an SIMD machine which can accommodate up to 8192 simple processing elements. It has an integrated memory which may be accessed in different modes by the processing elements, through a multistage switching network.

4.2.3 Array of Processors

These are SIMD machines with multiple arithmetic units operating in lockstep and performing the same operation on different data. The control unit broadcasts an instruction to be executed on local data by all processing elements. Each processing element can make minor modifications to the broadcast instruction (usually operand address modification) or be programmed to ignore the instruction. To achieve the potential parallelism of an array of processors, special attention must be paid to the data placement and algorithm design.

Machines built of processor arrays can be distinguished by the following characteristics:

- a) Number of processors;
- b) Number of memory banks;
- c) Form of communication network; and,
- d) Form of local and global control.

Examples of machines built of array of processors are provided by ILLIAC IV, ICL Distributed Array Processor DAP at Queen Mary College in London, and Goodyear Aerospace Massively Parallel Processor (MPP). The latter contains 16,384 single bit-serial processing elements connected in a square array (128×128) with strictly nearest neighbor connections. It has a speed of 430 MFLOPS for 32-bit floating-point addition and 216 MFLOPS for multiplication. The input/output operations can be done simultaneously with processing.

4.2.4 Data Flow Computers

Data flow computers represent a radical departure from the classical von Neumann architecture (control flow computers). In a data flow computer, an instruction is ready for execution when its operands arrive. There is no concept of a control unit or a program counter. Rather, the data flow concept is based on the dependency graph of a computation. The algorithm for performing a given computation is first written in a special programming language designed for data-flow applications. The program, which has the appearance of a directed graph, is then implemented directly by a network of hardware units corresponding to the graph. Operands flow to the functional units and operate whenever all the operands are present and the result can be accommodated. The output with new instruction packets is then forwarded to the next functional unit. Hence, data need not have any permanent residence. Since the dependency graph depicts all the parallelism of a computation whether or not the parallelism is regular, the data flow architecture offers a high potential for capturing parallelism.

Two major problems arise for data flow machines. The first problem is that conventional programming languages, which evolved from a basically von Neumann concept, are not well matched to data flow architectures. The second problem which is common to all highly-parallel machines, is that concurrency is limited by the communication network which routes results from processing elements. In an attempt to overcome the first problem, the programming language VAL was developed at the Massachusetts Institute of Technology to provide simple expression of all levels of concurrency. As for the remedy to the second problem, all the viable possibilities for data flow interconnection network are identical to those of conventional multiprocessor structures.

Despite the aforementioned problems, data flow structures have high potential for use as attached processors to general-purpose computers.

4.2.5 Multiple Processors

Multiple processors are MIMD machines in which each processor is fully programmable and can execute its own program. Various classifications exist for these machines according to interprocessor communication (including the data transfer mode and the interconnection topology) and control (see Ref. 18). Multiprocessor systems can provide, at most, a linear speedup factor over single processors. That is, n processors, at best, can perform the same task as a single processor in $1/n$ th the time. In practical applications, the actual speedup ranges from $0.3n$, for simple problems, to a lower bound of $\log_2 n$. The factors that prevent the realization of linear speedup include the following:

- a) *Synchronization.* Performance can be lost if the algorithm requires the processors to be periodically coordinated.
- b) *Algorithm.* An algorithm designed for a serial computer might not expose all the parallelism present in the problem. Also, a parallel algorithm might require more computations than its serial counterpart to solve the same problem.
- c) *Contention.* Multiple processors vying for the same resource (e.g., shared variable) can slow down the execution of individual processors.
- d) *Input/Output Operations.* Traditional I/O structures cannot feed a high performance processor fast enough to avoid processor idle time.

Examples of multiple processor machines are provided by the NASA Langley Finite Element Machine (FEM), and the CM* machine designed and built at Carnegie-Mellon University. The first will be discussed in the next section. The basic building block of the CM* machine is a processor-memory pair called a computer module, or Cm. The memory local to a processor is also the shared memory in the system. Up to 14 CM's are connected to a cluster. The cluster processors share a single bus and memory mapping processor. Clusters are connected via intercluster buses.

4.2.6 Other Highly-Parallel Architectures

Several highly-parallel architectures which do not fall into the previous categories have been designed and/or built. Most of these machines are research tools for advanced parallel architectures. A partial list of these machines is given subsequently (Ref. 14):

- a) *The Texas Reconfigurable Array Computer (TRAC)* built at the University of Texas at Austin. It currently has four processors and nine memories. However, the design can be easily scaled up to many processors and memories. Because it is based on dynamically coupled processors, input/output units, and memories in a variety of configurations, TRAC can be exploited to give computationally efficient parallel formulation of different algorithms, as well as of different phases within each algorithm. This is accomplished by changing the configuration to suit the structure of the algorithm.
- b) *The Blue CHiP (Configurable Highly-Parallel Computer)* designed and built at Purdue University in West Lafayette, Indiana. It consists of a collection of homogeneous processing elements (PEs) placed at regular intervals in a lattice of programmable switches.

Each PE is a computer with its own local memory. This architecture is targeted at wafer-scale implementation.

c) *The Cosmic Cube* is an experimental machine built at California Institute of Technology. It consists of 64 identical microprocessors (16-bit Intel 8086 and 8087 processors with 136K Bytes of memory) called nodes. Each of the nodes executes programs concurrently, and each can send messages to six other nodes (nearest-neighbor nodes) in a communication network based on a six-dimensional cube, or hypercube. A 1,024 node machine is planned using more modern microprocessors and a megabyte of memory per node. The resulting system is anticipated to have a peak performance equivalent to 50 times that of CRAY 1-S, and cost about \$2 million.

d) *The Ultracomputer* is a project at New York University which has, as yet, not been built. The highly-parallel machine aims to combine hundreds of thousands of small, relatively conventional processing elements, all using a large shared memory. Processors communicate with memory through a very high-bandwidth switching network which executes a few operations vital to very highly-parallel interprocess synchronization, in addition to its basic data-routing function.

4.2.7 Potential of Highly-Parallel Architectures for Computational Mechanics and Flight-Vehicle Structures Technology

In evaluating the impact of highly-parallel machines on flight-vehicle structures computations, it is useful to define two classes of problems and two distinct approaches to bringing the power of highly parallel machines to bear. The two classes of problems are:

a) *Large-Scale Complicated Problems.* These are problems that are much too large to fit on any commercially available computer (e.g., number of unknowns in excess of one billion). Examples of these problems are large, three-dimensional turbulent eddy simulations of aerodynamic flows, large nonlinear interaction problems, and large-scale multidisciplinary optimization problems.

b) *Medium-Scale Problems.* These are problems that can be solved on current super-systems (e.g., nonlinear problems with tens of thousands of unknowns). However, the cost of running them can be fairly high and, therefore, parametric studies cannot be made. Examples are provided by aerodynamic simulations of compressor- or turbine-blade flow fields and large nonlinear structural dynamic problems.

The most suitable approach for the first class of problems is based on using multiple processor machines with many processors. The interconnection between the processors must reflect both the geometry of the problem and the nature of the algorithms employed. A square grid of processors with only nearest-neighbor connections is the most obvious first choice.

The spatial allocation of processors involves mapping the different activities to be performed onto the processors of the system. With static mapping, the spatial allocation is fixed before the program execution begins. A variety of mapping schemes can be used such as one zone and/or one node per processor, or one row of nodes per processor. Adjacent zone computations can be mapped onto adjacent processors. As long as the size of the problem matches the size of the system, and the amount of computation per processor can be reasonably well equalized, this approach is quite attractive.

If, on the other hand, the program size and structure do not match the system size and structure, complete locality cannot be maintained. If resources are allocated dynamically, the assignment function must map operations which are likely to communicate only nearby processors. This would greatly complicate the resource allocation problem. The practical application of multiprocessor machines to large-scale problems requires the development of semiautomatic procedures to program a large amount of processors to perform similar but different tasks.

For the second class of problems, high throughput performance can be obtained by attaching multiple special-purpose functional units to a mainframe (or a supermini computer) through the use of a fast bus (or buses). These special-purpose functional units can be thought of as replacing many of the subroutines in a scientific software library. They could include linear algebraic systolic array modules, a sparse matrix processor, and an inner product chip.

The major problem in the effective use of this approach is communication between the different devices. Much numerical analysis must be rethought in this architectural context. For example, higher-order temporal integration schemes and higher-order techniques for solution of nonlinear equations, requiring less data to drive them (at a cost of increased arithmetic), might be more attractive than lower-order methods.

4.3 Special-Purpose Computing Hardware

With the continued reduction in the cost of computer hardware, a number of special-purpose computers have emerged that offer increased speed for specific problems, when

compared with general-purpose computers. An example of these special-purpose computers is the Finite Element Machine (FEM) being developed at NASA Langley Research Center. It is an MIMD computer used as an experimental research tool for demonstrating the potential of using highly-parallel architectures for high-speed yet low-cost solutions of finite element problems. Many aspects of finite element calculations are well suited to parallel computation (e.g., generation of elemental arrays, and iterative solution of global equations).

The NASA Langley Finite Element Machine, shown in Fig. 13, is designed to have 36 processors operating either asynchronously or synchronously, in parallel, and communicating via a local and a global bus. Currently, eight of the processors on the machine are operational. Each of the processors contain a 16-bit CPU, a special floating-point unit, 32K bytes of local RAM and 32K bytes of ROM. The ROM contains a special parallel operating system, termed NODAL EXEC together with a library of frequently-used Pascal-callable routines, termed PSLIB. Figure 13 shows the cabinet that will house the 36 processors and a typical 16-bit CPU board (one of 108 boards). The controller of the array of processors is a minicomputer with 116M Bytes of disk storage, tape drive, and printer. The controller is accessed via terminals (both text and graphics) and can transfer data to and from a DEC VAX 11/780 superminicomputer. The strategy used for exploiting parallelism using the first eight processors of FEM (see Ref. 27) is to perform all the computations in parallel on the array of processors. In Ref. 28 the application of iterative algorithms to the solution of large sparse finite element equations on the finite element machine is examined.

The architecture of the FEM provides for nearest-neighbor connection as well as global connection which appears to have high potential for large-scale finite element applications including steady-state and transient response problems.

Recently other studies based on the Finite Element Machine concept have been made (see Refs. 29, 30 and 31). In Ref. 29 the Japanese PAX-128 parallel computer is described. It is an MIMD microprocessor array with 128 8-bit processing elements arranged in a two-dimensional network with end-around and nearest-neighbor-mesh connection, and data broadcasting bus line. The system has three basic components: host computer, control unit and processing unit, and has been used for solution of linear algebraic equations and partial differential equations. In Ref. 30 the application of a Finite Element Machine concept to the generation and assembly of finite element stiffness matrices is described. The hardware used consisted of an IEEE-696 S-100 bus microcomputer with a 16-bit CPU and a master processor which controls and coordinates the activities of 8086/8087 VLSI chip set processors working in parallel. In Ref. 31 several parallel algorithms for the solution of the finite element equations are examined. On the basis of the results shown in Ref. 31, it appears that for computers with unlimited number of processors, the direct cyclic reduction algorithm is best suited, and for machines with limited number of parallel processors, both the direct Gauss factorization and the Jacobi-like iterative methods rank favorably.

4.4 Small Systems

To date a broad spectrum of low-cost small systems exist including handheld computers (the micro micros - see Ref. 32), desktop computers, engineering workstations, minicomputers and superminis. Several classifications have been attempted for these systems based on word length (8-bit, 16-bit and 32-bit machines), cost, amount of directly addressable memory, and computing speed (see, for example, Ref. 33). However, the dramatic increase in hardware capabilities of small systems coupled with the rapid reduction in cost make these classifications of questionable value. Herein, a brief discussion is given of the impact of minicomputers and microcomputers on computational mechanics and flight-vehicle structures technology.

4.4.1 Minicomputers and Superminis

Minicomputers were introduced in the late 1960's as a new concept in decentralized computing. At first minicomputers were used only for pre- and postprocessing of finite element calculations. However, as minicomputers with larger address spaces and faster arithmetic appeared, a migration of engineering analysis and finite element software to minicomputers started to take place, and with the emergence of the superminis having 32-bit word length, very large address spaces, and fast hardware or firmware for floating point operations (e.g., the DEC VAX 11/780, the PRIME 850, ECLIPSE MV/10000, IBM 4341, and Norsk Data 570), many of the large-scale finite element programs (e.g., ASKA, ANSYS, MARC, MSC/NASTRAN, and SPAR (predecessor of EISI-EAL)), were installed on the minicomputer (see, for example, Ref. 34).

Recently, a new generation of high-performance superminis has been developed which outspeeds conventional superminis by a factor of 2. Examples of the new computers are provided by the PYRAMID 90X and the RIDGE 32 computers. The high performance of these machines is attributed to the use of Reduced-Instruction-Set-Computer architecture RISC with large register stacks organized as overlapping register sets. In the RISC design all operations except memory loads and stores are register-to-register. Overlapped register sets minimize data movement in procedure calls and allow parameter passing through "windows" in the overlapping register sets.

Minicomputers provide user-friendly operational characteristics including interactive operating systems, local control over resources and turnaround, and high-speed graphics. However, currently available minicomputers (including the superminis) are considerably slower than mainframes, and therefore, their effectiveness is limited in solving large-scale finite element problems such as those encountered in crash dynamics and three dimensional fluid-flow simulations (see Refs. 35 and 36).

4.4.2 Microcomputers and Engineering Workstations

There now exists a large number of microcomputers and desktop computers, both portable and stationary. A partial list of the desktop computers and some of their characteristics is given in Ref. 37. The first generation of microcomputers had 8-bit processor chips and 64K bytes of RAM (a few 8-bit designs have increased the addressable RAM to 256K bytes). The second-generation machines are 16-bit machines with up to 1M byte of RAM. They generally use one of three CPU chips (Ref. 38): the Intel 8088, Intel 8086, or the Motorola 86000. The Intel 8088 uses 16-bit architecture internally, but handles data on an 8-bit bus. By contrast, the 8086 is a 16-bit processor and, therefore, it is more powerful and faster because it transfers more data at one time. The Motorola 68000 which is the most powerful of the three, takes data on either a 16-bit or 32-bit bus but does its processing internally 32 bits at a time. The third-generation microcomputers, termed *intelligent engineering workstations*, have 32-bit processor chips and over 16M bytes (or more) of addressable memory, high-resolution graphics (1000 x 1000 pixels), multitasking, multiwindowing, and are likely to replace current superminis. Examples of such systems are Apollo, HP-9000, and IBM 9000 computers.

A number of large-scale finite element programs (e.g., ANSYS, EISI-EAL and MSC/NASTRAN) have been installed on the Apollo computer. Also, several small finite element programs have been developed for the microcomputers (see Refs. 39 to 43). If the program is segmented in order to reduce high-speed memory requirements, fairly large finite element problems can be solved on the microcomputer (see Ref. 39). However, since many large problems require 64-bit accuracy, it could be slow to run them on microcomputers with 8-bit or even 16-bit processors. The availability of 32-bit microcomputers such as the Apollo, HP-9000 and IBM 9000 computers is likely to extend the range of finite element problems that can be solved by the microcomputer. Also, microcomputers are likely to become an integral part of laboratory testing for processing data in support of direct flight-vehicle certification.

4.5 Attached Processors

For cost-effective computing a fast backend computer can be attached to a slower host minicomputer to achieve computational speeds comparable to those of large mainframes at a fraction of the cost of the mainframes. The fast computer is referred to as the attached processor. The dual-processor system consisting of the minicomputer and the attached processor can be used to extend the range of problems that can be solved effectively on the minicomputer. The dual-processor arrangement is particularly useful for a small user community. Examples of attached processors include the floating point systems FPS AP-120B, AP-190L, and FPS-164, the CSPI MAP-300 and MAP-6400 series, and the recently announced ST-100 attached processor of the Star Technologies, Inc. While the FPS and ST-100 processors have synchronous architecture, the CSPI processors have asynchronous multiple-bus architecture. Each of the FPS attached processors has the following features (Ref. 11):

- a) It is designed and accessed as a peripheral (i.e., like a tape drive) for a conventional host minicomputer, and it is intended to enhance the performance of the host in specific numerical computing tasks.
- b) It achieves high performance through both parallelism and pipelining.
- c) It includes an arithmetic section containing one adder and one multiplier capable of operating in parallel.
- d) It is not hardwired. Rather, it can be programmed by the user in FORTRAN or assembler language to perform a variety of computational tasks.

Parallelism is a major feature of the FPS attached processors that allows high-speed vector and matrix processing. In addition, the parallel structure of the attached processors allows the overhead of loop indexing, array indexing, and data fetching to be performed in parallel with floating-point computations. Parallel operations include integer indexing, branch instructions, memory fetches, memory storage, I/O with the host, and floating-point arithmetic. These features make the FPS systems much faster processors than most general-purpose sequential computers. The peak processing rate of the FPS attached processors is 12 MFLOPS and a concurrent six-million integer and addressing operations per second. However, such performance is difficult to realize in practical applications.

The ST-100 attached processor with planned delivery in 1983 is advertised to have a peak performance of 100 MFLOPS. It has the following characteristics:

- a) Multiple programming internal processors and multiple arithmetic units.
- b) High-speed logic circuitry.
- c) Main memory of 32-bit words which can be expanded to 8 M words.
- d) 48K words of random access data cache memory.
- e) Multiple host interfaces.

Software for attached processors can be developed on three different levels:

- a) FORTRAN high-level language with many specialized functions;
- b) Assembly language macros using a library of routines provided by the manufacturer; and,
- c) Pure micro-assembly language at the machine level.

Recently, the software support capabilities of the attached processors have considerably improved. Such support software includes an ANSI FORTRAN-77 compiler, assembler, loader, object code linker, and simulator - all running on the host computer.

Recent studies (e.g., Refs. 44, 45 and 46) have shown that the combination of a minicomputer and an attached processor in a dual-processor system can extend the range of finite element problems that can be solved effectively by the minicomputer. This capability is accomplished through efficient implementation of both the numerical algorithms used in the finite element solution and the communication protocol between the host minicomputer and the array processor. The question of synchronization between the different parts of the system to ensure correct and efficient function of the minicomputer/attached processor system is of paramount importance.

5. ADVANCES IN SPECIAL ALGORITHMS AND PROGRAMMING LANGUAGES FOR NEW COMPUTING SYSTEMS

The advent of new high-performance computing systems also introduces the problems of designing (a) numerical algorithms that execute efficiently on them, and (b) software with which these algorithms can be expressed. The development of parallel numerical algorithms for pipeline and parallel computers has received considerable attention since the early sixties even before the implementation of these architectures. A number of survey papers have been written on the subject (see, for example, Refs. 47 to 49). These studies have demonstrated that severalfold increases in speed over conventional sequential machines can be obtained if the calculations are arranged to take advantage of the specific hardware.

In this section a brief review is given of the recent progress made in special numerical algorithms and programming languages that are likely to impact finite element technology.

5.1 Parallel Numerical Algorithms

In parallel algorithms, independent computations are performed in parallel (i.e., executed simultaneously (or concurrently)). To achieve this parallelism, the algorithm is divided into a collection of independent tasks (or task modules) which can be executed in parallel and which communicate with each other during the execution of the algorithms. Parallel algorithms can be characterized by the following three factors (see Ref. 23):

- a) Maximum amount of computation a typical task module can do before having to communicate with other modules;
- b) Intermodule communication topology, which is the geometric layout of the network of task modules; and,
- c) Executive control to schedule, enforce the interactions among the different task modules, and ensure the correctness of the parallel algorithm.

The aforementioned three factors have been used in Ref. 21 as a basis for classifying parallel algorithms on a conceptual level, and for relating each parallel algorithm to the parallel (or pipeline) architecture to which it naturally corresponds.

The design of a parallel algorithm must deal with a host of complex problems, including data manipulation, storage allocation, memory interference, and in the case of parallel processors, interprocessor communication.

5.2 Matching Numerical Algorithms with Computer Architectures

In order to obtain the optimum performance from any computing system it is necessary to either tailor the numerical algorithm to suit the architecture of the computer, or to select the architecture which will effectively execute the numerical algorithm. The numerical algorithms developed for synchronous, asynchronous parallel computers and systolic machines are discussed subsequently.

5.2.1 Algorithms for Synchronous Parallel Computers (MISD and SIMD Machines)

The introduction of parallelism in the numerical algorithm used on synchronous parallel computers is usually referred to as *vectorization*. The vectorization of any mathematical operation is dependent on the particular hardware. As an example, for a task to be vectorizable on pipeline computers, it should contain three characteristics: 1) repeated operations; 2) independence of each result from the others; and 3) the members of each operand are packed in either contiguous memory locations or at fixed intervals from each other (with some restrictions in certain hardware configurations). Efficiency and effectiveness of vectorized numerical algorithms on pipeline computers are affected by both the startup time and the average vector length.

Vectorized algorithms have been widely studied for a variety of applications and a variety of hardware configurations. Examples of successful vectorized algorithms that have been developed for synchronous computers include matrix operations, direct and iterative methods of solution of algebraic equations, eigenvalue extraction techniques, multigrid finite difference methods for solution of partial differential equations, and evaluation of elemental matrices for higher-order finite elements (see Refs. 50 to 53). Also, the development of parallel algorithms for the solution of partial differential equations on associative processors has been discussed in Ref. 54.

5.2.2 Algorithms for Asynchronous Parallel Computers (MIMD Machines)

In comparison with the parallel algorithms for synchronous computers, only a few studies have been made on parallel algorithms for asynchronous computers. Of particular interest are the asynchronous parallel iterative methods in which processors are not synchronized at all. In a truly asynchronous iterative algorithm, a process keeps computing iterates by using whatever information is currently available and releases immediately its computed results to other processes. Thus, the actual iterates generated by the method depend on the relative speed of the processes. The philosophy of "not waiting for other processes to complete their tasks" has been found to be a useful criterion for developing efficient numerical algorithms for asynchronous multiprocessors (Ref. 24). A typical approach to achieve this goal is to use copies. After a process completes its current task, it immediately starts working on a copy of the most recent global data.

5.2.3 Algorithms for Systolic Machines

Algorithms for systolic machines are designed for direct hardware implementation and, therefore, their task modules are simple and the interactions among them are frequent.

5.3 Performance Evaluation of Parallel Algorithms

It is important to make a distinction between two steps in evaluating the performance of new computing systems.

Comparative Performance. The ultimate cost-effective judgment that must be rendered in comparing new machines on a code can be a complicated function of architecture-related considerations. Moreover, the performance is so dependent on idiosyncrasies of the code and the compiler that such comparisons have a temporary value at best.

Single Machine Performance. The comparison of different algorithms and codings on a parallel machine is difficult to make when both parallel algorithm development (a general mathematical concept) and the control of data flow between parallel functional units and hierarchically-organized memories (a machine-dependent coding concept) are involved.

A number of measures have been proposed to quantify the performance of a computer program. One of the simplest, though inadequate measures is based on the CPU time consumed during the execution of the program (see Ref. 11). The two most commonly-used measures for the performance of parallel algorithms on multiple processor and array of processor type machines are based on: the ratio of the speedup over a single processor; and the ratio of the theoretical number of computation steps to the actual number of parallel steps. The first measure is suitable for machines with few processors. The second measure is used for machines with many processors and is strongly dependent on the complexity of the algorithm.

5.4 Software for Parallel Processing

The necessary software for supporting parallel processing includes special programming languages, data management systems and operating systems. Parallel programming is significantly more complicated than sequential programming because:

- a) It is difficult to keep track of several simultaneously occurring events.
- b) The potentially complex (time-dependent) interactions of the systems' components may result in elusive and nonrepeatable bugs.

There is a wide spectrum of language possibilities for improving the performance through parallel execution. Examples of the parallel processing languages are provided by Ada (Ref. 55) and Concurrent Pascal (Ref. 56). However, the suitability of these languages for multiple processor machines has not been investigated. Data management in highly-parallel systems can be a serious problem. The tasks performed by the data management system include supplying input to and analyzing output from the processors, and following a trace of the execution.

The operating systems for highly-parallel machines must be considerably more sophisticated than those for conventional machines. In addition to the usual issues encountered with conventional machines, a number of new issues have to be addressed including physical distribution of tasks among different processors, resource sharing, and interprocessor communication. This is still a research area, and to-date no operating systems have been developed for highly-parallel machines.

The absence of software and languages for highly-parallel machines are severe barriers to progress, since without them assessment of the performance of these machines would be very difficult to make.

6. SCENARIO FOR FUTURE ENGINEERING ANALYSIS SYSTEMS

The changes in the computing environment and capabilities discussed in the previous sections suggest significant changes in engineering analysis systems for the future. This section projects a scenario for a future engineering analysis environment, the attendant computer hardware, software, and numerical algorithms together with implications for needed research.

6.1 Future Engineering Analysis Environment

With the explosion in computer hardware alternatives as well as the maturing of the finite element method as a basic general-purpose numerical analysis tool competitive with other numerical techniques, the engineering analysis environment in the next five to ten years should be quite different from today. The basic ingredients today evolve primarily around a standard finite element or other numerical discretization technique implemented on a specific computer. The future environment should have several characteristics which are already becoming visible, and will be strongly oriented towards finite element formulations for approximation techniques and information management.

- a) Comparable finite element and other numerical discretization capabilities will be available to the user on an array of different levels of processors from desktop (personal) computer to supercomputer. Each level will have features from which the user can select.
- b) Finite element software and firmware (i.e., software on ROM) will exist which ranges from stand alone systems to major subsystems in large integrated CAD/CAM systems.
- c) Hardware and software will be distributed and networked so that one can utilize combinations of functions distributed over different processors to maximize the advantages of each level of computation.
- d) Arrays of user interface facilities will continue to grow to the level where modeling, mesh generation, user options, tutorials and other features will be so tailored that a user can be led through program use. Results will be readily presented in a variety of multicolored graphical displays. The simplified user interfaces typified by today's video games and cockpit simulations will begin to show up in engineering analysis capabilities. Even some of the smart aids evolving from artificial intelligence (knowledge-based and expert systems) and optimization theory will provide assistance to the user in decisionmaking.
- e) Algorithms will be expanded in scope and capability for numerical solutions and engineering applications. The increased arrays of hardware available will require expansion of algorithm alternatives to exploit these hardware changes. Algorithms chosen will range from those appropriate for small SIMD hardware to those which exploit large highly-parallel MIMD architectures.
- f) Specialized hardware/firmware devices will exist which are tailored to carrying out specific finite element or other engineering analysis tasks. Systolic machines,

data flow machines, and chips will exist which generate discretized structure matrices, assemble these matrices, solve equations, or provide total engineering analysis solutions. Thus the counterpart of electronic "blackboxes" which exist in applications of control theory (e.g., aircraft) will show up as engineering analysis capabilities.

6.2 Future Computer Hardware Environment

The future computer hardware environment available to a user will be characterized by an array of computer hardware capabilities in an expanding level of facilities. Figure 14 shows one approach to visualizing these facilities. The hardware will range from a core capability of supercomputers to provide large-scale computations outward to a periphery which contains a wide variety of intelligent engineering workstations well-suited for interactive user interface/control and moderate scale calculations. The number of alternatives for the central core computers will be relatively few whereas the workstation alternatives will be virtually unlimited. In some cases a computer facility may be just an array of networked workstations.

In the context of finite element applications, for example, the interactive model generation and evaluation of results (in graphic form) can be done on the personal computers, and the solution of the global equations can be performed on the supercomputer. Intermediate calculations (e.g., element generation, assembly of elemental matrices, and computation of secondary fields) could be done on the minicomputers or on the personal computers, depending on the instantaneous load on the system.

The organization of this hardware will require:

- a) Extensive facilities for local and long-range networking to make all hardware and attendant software readily available to each user.
- b) Development of demand-sensitive load-sharing systems to allow programs to migrate from one hardware/operating environment to another. Figure 15 illustrates a typical arrangement of networked hardware composed of a loosely coupled very high-speed network to interface the minicomputers and the large-scale computers, and a moderate-speed local area network to connect workstations. Local networking can facilitate cooperative multidisciplinary investigations and design projects among team members interacting through shared databases. Inexpensive long-range networking can have several significant effects. Issues of proprietary data can be reduced by sending the program to the location of the owner of the proprietary data, performing the computations there, and receiving only the processed nonproprietary results.

6.3 Future Engineering Software Systems

The organization of a future engineering software system will change as a result of the growing maturity of CAD/CAM systems and the recognition by computer vendors of the importance of supporting such software as a major business thrust. Instead of having an operating system and support software developed primarily for business applications, the vendor supported facilities will be tailored for CAD/CAM needs. The total engineering software system available to the user (see Fig. 16) can be conveniently visualized as an expanding level of capabilities and specializations emanating from a core containing a user-friendly operating system through a software "infrastructure" containing the engineering support software, and a periphery containing a large array of application programs. A key element of the future will be the inclusion of the basic finite element and matrix handling software as part of the software infrastructure which is incorporated as part of system software rather than as application programs. The data management and utilities will provide multilevel integration of various applications for multidisciplinary calculations.

The organization of a future engineering analysis system will be compatible with this concept and will exploit the data management and other features of the infrastructure. One approach to the organization of a future finite element system is shown in Fig. 17. This figure shows the basic finite element and matrix handling software (such as data management, control, equation solving, etc.) as part of the software infrastructure and the discipline specifics (such as element properties) as part of the application software. Thus, the infrastructure provides the information manipulation and handling capabilities, and the applications layer contains the fundamental engineering theory and assumptions.

Of the major database models, the relational database management system (see, for example, Ref. 57) appears to be the most suitable for integration with finite element databases because of its formalism and higher level of physical and logical data independence. Research towards more advanced engineering database concepts such as the IPAD multischema approach (Ref. 58) should continue to expand the information management facilities available to support engineering analysis and design.

6.4 Future Numerical Algorithms

Numerical algorithms available in the software for solution to engineering analysis problems will expand significantly beyond today's relatively stable set. This expansion will be due to the many available alternatives in hardware as well as the expanding capabilities of the software infrastructure. The core of the numerical algorithms available to the analyst will continue to be the strategy used to solve simultaneous linear algebraic equations and manipulate matrices (see Fig. 18 for one visualization). The next layer will be the specific classes of applications and the outer periphery will contain a wide variety of support algorithms and techniques to provide such functions as modeling, accuracy improvement and computational speedup.

One hierarchical approach to organization of solution algorithms for finite element related formulations is shown in Fig. 19. At the top level are the matrix processors and the linear equation solvers. The second level contains the specific applications such as linear and nonlinear steady-state analysis, eigenvalue analysis and transient response analysis. The third level contains the wide array of algorithms for specific tasks (e.g., generation of finite element arrays for a variety of applications). In the application of the total set of algorithms, control software provides the user with facilities for automatic or user-directed selection of appropriate combinations of algorithms for this specific application.

6.5 Future Research Areas

Research is needed in a number of areas to support effective extensions of today's engineering analysis capabilities into the future environment (next five to ten years) characterized here. The two key elements of the future are *explosion of alternatives and planning for changes*. Research needs include:

- a) Engineering Research
 - o Identification of requirements for a future engineering analysis system.
 - o Modeling strategies for various disciplines and physical problems.
 - o Continued advancements in analytical capabilities to simulate the response of complex engineering systems.
- b) Engineering/Computer Science Research Tailored to Finite Element Requirements
 - o Distributed processing.
 - o High-speed communications and networking.
 - o Control of distributed activities.
 - o Information management.
 - o Expert system concepts and applications.
 - o Three-dimensional geometry modeling and display.
 - o Software design of future engineering analysis systems.
 - o Development/transfer of algorithms to microprocessor implementation.
- c) Engineering/Management Research for Engineering Analysis Tasks
 - o Managing strategies for using hardware/software capabilities.
 - o Matching hardware/software alternatives to achieve cost effective results.
 - o Standards for integrating distributed engineering analysis systems.
 - o Selection/management/control of analysis capabilities to solve new complex problems.
- d) Numerical Algorithms
 - o Identification and development of methods for exploiting parallelism in the calculations.
 - o Methods for decomposition of calculations into arrays of different processors.
 - o Development of firmware algorithms for special tasks.

6.6 Summary of Changes for Future Engineering Analysis Methodology

In summary, the scenario of future engineering analysis systems will encompass more choices and options for solution strategies using hardware, software and algorithms. There will be arrays of hardware, different levels of computational capabilities, large arrays of software packages, work distributed across many processors and packages of capability on specific processors. Software and applications interface standards, information management and control procedures will become increasingly important. Analysis research and development must become multidisciplinary, blending the combined technologies of computer hardware/software, engineering, mathematics and even management. Applications will require better management strategies together with increased standardization to maximize efficiency and minimize cost. A key ingredient of future engineering analysis development and application will be the management of change and considerations of alternatives rather than the current approach of addressing technical advances in a relatively stable and structured hardware/software environment.

7. SUMMARY AND CONCLUDING REMARKS

A detailed review is given of the recent advances in computer technology that are likely to impact finite element computations. The characteristics of new and projected computing systems are summarized. At one end of the spectrum there are the large super-systems such as the CRAY-1S and the CDC CYBER 205. The performance of supersystems will continue to improve and their speed is likely to reach 20 GigaFLOPS before the end of the present decade. These supersystems will make possible new levels of sophistication in analytical modeling as well as in problem depth and scope which were not possible before. At the other end of the spectrum the small low-cost computer systems, particularly the microcomputers (handheld computers) and the desktop computers (engineering workstations) will provide a high degree of interactivity and free the analysts from constraints that are often imposed on them by large centralized computation centers. To improve the performance of the small systems, attached processors (some on single chips) will be combined with them to form high-performance dual-processor systems.

The special-purpose highly-parallel systems will occupy an intermediate position between the two extremes (the supersystems and the small systems). Special-purpose processors such as the Finite Element Machine, Navier-Stokes machines and Partial Differential Equation (PDE) machines will continue to be built for fast solution of engineering problems. However, their effectiveness will depend on the availability of software to exploit the parallelism in these processors (e.g., programming languages, operating systems and data management systems).

A scenario is presented for future hardware/software environment and engineering analysis systems. In this scenario a broad spectrum of analysis software and firmware is projected ranging from stand-alone systems to modules in large integrated CAD/CAM systems. The user interface facilities (including geometric modeling, mesh generation, input facilities, graphics displays, etc.) will continue to improve. The networked computer hardware/software environment of the future will be more complex and offer many more choices to the user than are available today. This environment provides an opportunity for some very creative research on management-type methods for selecting the best combinations of hardware and software for cost-effective solutions to problems. It is likely that the selection of the appropriate analysis software, firmware and hardware for the solution of an engineering problem will be done with the aid of artificial intelligence-based expert systems.

The discussion of the new computing systems presented herein is intended to give engineering analysts some insight into the potential of these systems for providing cost-effective solutions of complex engineering problems, and to stimulate research and development of the necessary numerical algorithms, firmware and software to realize this potential. The future offers exciting opportunities for new and innovative analytical research in exploiting the new computing environment.

REFERENCES

1. Schrem, E., "Finite Element Software in the Next Decade," Proceedings of the World Congress on Finite Element Methods in Structural Mechanics, Bournemouth, Oct. 12-17, 1975.
2. Fenves, S. J., "Future Directions of Structural Engineering Applications," Computers and Structures, Vol. 10, Nos. 1/2, April 1979, pp. 3-5.
3. MacNeal, R. H., "One Man's View of Trends in Finite Element Analysis," in Computational Aspects of the Finite Element Method, ed. by J. F. Gloudeman and R. E. Rosanoff, Bundesanstalt fur Materialprufung, Berlin, 1979, pp. 1-19.
4. Swanson, J. A., "Present Trends in Computerized Structural Analysis," Computers and Structures, Vol. 10, Nos. 1/2, 1979, pp. 33-37.
5. Stevenson, D., "New Computing Systems and Their Impact on Computation," Supplement to NASA CP-2147, 1981, pp. 1-6.
6. Swanson, J. A., "Opportunities Provided by Large Computers," in New and Future Developments in Commercial Finite Element Methods, ed. by J. Robinson, Robinson and Associates, Dorset, England, 1981, pp. 436-450.

7. MacNeal, R. H., "Trends in Finite Element Structural Analysis," presented at the AIAA/ASME/ASCE/AHS 23rd Structures, Structural Dynamics and Materials Conference, New Orleans, LA, May 10-12, 1982.
8. Lykos, P. and White, J., "An Assessment of Future Computer System Needs for Large-Scale Computation," NASA TM-78613, 1980.
9. Bloch, E. and Galage, D. J., "Component Progress: Its Effect on High Speed Computer Architecture and Machine Organization," in High Speed Computer and Algorithm Organization, ed. by D. J. Kuck, D. H. Lawrie and A. H. Sameh, Academic Press, New York, 1977, pp. 13-39.
10. Chapman, D. R., "Computational Aerodynamics - Development and Outlook," AIAA Journal, Vol. 17, No. 12, Dec. 1979, pp. 1293-1313.
11. Hockney, R. W. and Jesshope, C. R., Parallel Computing, Adam Hilger, Inc., Bristol, 1981.
12. Reisman, A., "Device, Circuit and Technology Scaling to Micron and Submicron Dimensions," Proceedings of the IEEE, Vol. 71, No. 5, May 1983.
13. Swartzlander, E. E. and Gilbert, B. K., "Supersystems: Technology and Architecture," IEEE Transactions on Computers, Vol. C-31, No. 5, May 1982, pp. 399-409.
14. Manuel, T., "Parallel Processing," Electronics, June 16, 1983, pp. 105-114.
15. Groves, B., "Getting VHSIC Into Real-World Systems," Defense Electronics, Vol. 15, No. 1, Jan. 1983, pp. 102-111.
16. Flynn, M. J., "Some Computer Organizations and Their Effectiveness," IEEE Transactions on Computers, Vol. C-21, 1972, pp. 948-960.
17. Ostlund, N. S., Hibbard, P. G., and Whiteside, R. A., "A Case Study in the Application of a Tightly Coupled Multiprocessor to Scientific Computations," in Parallel Computations, ed. by G. Rodrigue, Academic Press, 1982, pp. 315-364.
18. Fathi, E. T. and Krieger, M., "Multiple Microprocessor Systems: What, Why and When," Computer, Vol. 16, No. 3, March 1983, pp. 23-32.
19. Infotech State-of-the-Art Report "Supercomputers," Vols. 1 and 2, Infotech Inter., Ltd., Maidenhead, Berkshire, England, 1979.
20. Feigenbaum, E. A. and McCorduck, P., The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge, Addison-Wesley Publ. Co., 1983.
21. Oakes, W. R. and Browning, R. V., "Experiences Running ADINA on CRAY-1," Proceedings of the ADINA Conference, Aug. 1979, Rep. 82448-9, Massachusetts Institute of Technology, Cambridge, pp. 27-42.
22. Haynes, L. S., Lau, R. L., Siewiorek, D. P., and Mizell, D. W., "A Survey of Highly-Parallel Computing," Computer, Jan. 1982, pp. 9-24.
23. Kung, H. T., "Why Systolic Architectures," Computer, Jan. 1982, pp. 37-46.
24. Kung, H. T., "The Structure of Parallel Algorithms," Advances in Computers, Vol. 19, Academic Press, 1980, pp. 65-112.
25. Law, K. H., "Systolic Schemes for Finite Element Methods," Rep. R-82-139, Dept. of Civil Engineering, Carnegie-Mellon University, July 29, 1982.
26. Melhem, R., "Formal Verification of a Systolic System for Finite Element Stiffness Matrices," Tech. Rep. ICMA-83-56, Institute for Computational Mathematics and Applications, University of Pittsburgh, PA, 1983.
27. Storaasli, O. O., Peebles, S. W., Crockett, T. W., Knott, J. D., and Adams, L., "The Finite Element Machine: An Experiment in Parallel Processing," NASA TM-84514, NASA Langley Research Center, Hampton, VA, July 1982.
28. Adams, L. M., "Iterative Algorithms for Large Sparse Linear Systems on Parallel Computers," NASA CR-166027, NASA Langley Research Center, Hampton, VA, Nov. 1982.
29. Hoshino, Tsutomu, et al., "Highly Parallel Processor Array 'PAX' for Wide Scientific Applications," Proceedings of the 1983 International Conference on Parallel Processing, Aug. 23-26, 1983, Columbus, OH, publ. by IEEE Computer Society Press, Silver Spring, MD, pp. 95-105.
30. Salama, M., Utku, S., and Melosh R., "Parallel Solution of Finite Element Equations," Proceedings of the Eighth ASCE Conference on Electronic Computation, University of Houston, Houston, TX, Feb. 21-23, 1983, pp. 526-539.
31. McGregor, J. and Salama, M., "Finite Element Computation with Parallel VLSI," Proceedings of the Eighth ASCE Conference on Electronic Computation, University of Houston, Houston, TX, Feb. 21-23, 1983, pp. 540-553.
32. Tyler, M., "The Micro Micros," Datamation, Vol. 28, No. 13, Dec. 1982, pp. 34-42.
33. Conaway, J. H., "Structural Engineering Software on Small Computers," American Society of Mechanical Engineers, Paper 80-C2/Aero-7, Aug. 1980.
34. Foster, E. P. and Storaasli, O. O., "Structural Analysis on a Minicomputer," Engineering Software, ed. by R. A. Adey, Pentech Press, London, 1979, pp. 43-54.
35. Storaasli, O. O. and Murphy, R. C., "Finite Element Analysis in a Minicomputer/Mainframe Environment," in Research in Computerized Structural Analysis and Design, NASA CP-2059, 1978, pp. 77-83.
36. Wilson, E. L., "The Use of Minicomputers in Structural Analysis," Proceedings of the ADINA Conference, Aug. 1979, Rep. 82448-9, Massachusetts Institute of Technology, Cambridge, MA, pp. 293-303.
37. Edwards, J., "The Desktop Buyer's Guide to Small Computers," Desktop Computing, April 1983, pp. 52-69.
38. Lu, C., "Microcomputers - The Second Wave," High Technology, Sept/Oct 1982, pp. 36-52.
39. Beckx, E., Rammant, J. P., and Schymkowitz, "Case Study on Desk Computer: Finite Element Analysis in BASIC," Finite Element Methods in the Commercial Environment, ed. by J. Robinson and publ. by Robinson and Associates, Dorset, England, 1978, Vol. 2, pp. 670-780.
40. Firmin, A., "FESDEC - A Finite Element System for Desktop Computers," Finite Element News, Jan. 1980, pp. 22-23.
41. Wilson, E. L., "SAP-80 - Structural Analysis Programs for Small or Large Computer Systems," presented at CEPA 1980 Fall Conference and Annual Meeting, Newport Beach, CA, Oct. 13-15, 1980.

42. Kelly, P. and Watson, A. S., "Structural Analysis Using Desktop Computers," in Engineering Software II - Proceedings of the Second International Conference on Engineering Software, held at Imperial College of Science and Technology, London, March 1981, CML Pubs., Ltd., Southampton, 1981, pp. 346-359.

43. Griffen, O. H. and Wilson, C. R., "Finite Element Analysis on a Microprocessor-Based Personal Workstation," Tech. Note, Computers and Structures, Vol. 17, No. 4, 1983, pp. 617-619.

44. Strohkorb, G. A. and Noor, A. K., "Potential of Minicomputer/Array Processor System for Nonlinear Finite Element Analysis," NASA TM-84566, June 1983. Also publ. in Computers and Structures, Vol. 18, No. 4, 1984, pp. 703-718.

45. Swanson, J. A., Cameron, G. R., and Haberland, J. C., "Adapting the ANSYS Finite Element Program to an Attached Processor," Computer, Vol. 16, No. 6, June 1983, pp. 85-91.

46. Thompkins, W. T., Jr. and Haimes, R., "A Minicomputer/Array Processor/Memory System for Large-Scale Fluid Dynamic Calculations," in Impact of New Computing Systems on Computational Mechanics, ASME, Nov. 1983, pp. 117-126.

47. Miranker, W. L., "A Survey of Parallelism in Numerical Analysis," SIAM Review, Vol. 13, No. 4, Oct. 1971, pp. 524-547.

48. Sameh, A. H., "Numerical Parallel Algorithms - A Survey," in High-Speed Computer and Algorithm Organization, Academic Press, 1977, pp. 217-228.

49. Heller, D., "A Survey of Parallel Algorithms in Numerical Linear Algebra," SIAM Review, Vol. 20, No. 4, Oct. 1978, pp. 740-777.

50. Voigt, R. G., "The Influence of Vector Computer Architecture on Numerical Algorithms," in High-Speed Computer and Algorithm Organization, Academic Press, 1977, pp. 229-244.

51. Ortega, J. M. and Voigt, R. G., "Solution of Partial Differential Equations on Vector Computers," Proceedings of the 1977 Army Numerical Analysis and Computers Conference, ARO Rep. 77-3, pp. 475-525.

52. Rodrigue, G. (ed.), Parallel Computations, Academic Press, 1982.

53. Noor, A. K. and Hartley, S. J., "Evaluation of Element Stiffness Matrices on CDC STAR-100 Computer," Computers and Structures, Vol. 9, No. 2, 1978, pp. 151-161.

54. Gilmore, P. A., "Numerical Solution of Partial Differential Equations by Associative Processing," Proceedings of the 1971 Fall Joint Computer Conference, AFIPS, Vol. 39, pp. 411-418.

55. Brender, R. F. and Massi, I. R., "What is Ada?," Computer, Vol. 14, No. 6, June 1981, pp. 17-22, 24.

56. Coleman, D., Gallimore, R. M., Hughes, J. W., and Powell, M. S., "An Assessment of Concurrent Pascal," Software - Practice and Experiment, Vol. 9, No. 10, Oct. 1979, pp. 827-837.

57. RIM User's Guide, Academic Computer Center, University of Washington, W33, Jan. 1980.

58. Fulton, R. E., "CAD/CAM Approach to Improving Industry Productivity Gathers Momentum," Astronautics and Aeronautics, Vol. 2, Feb. 1982, pp. 64-70.

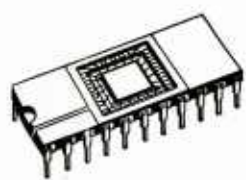


Fig. 1 - Silicon chip.

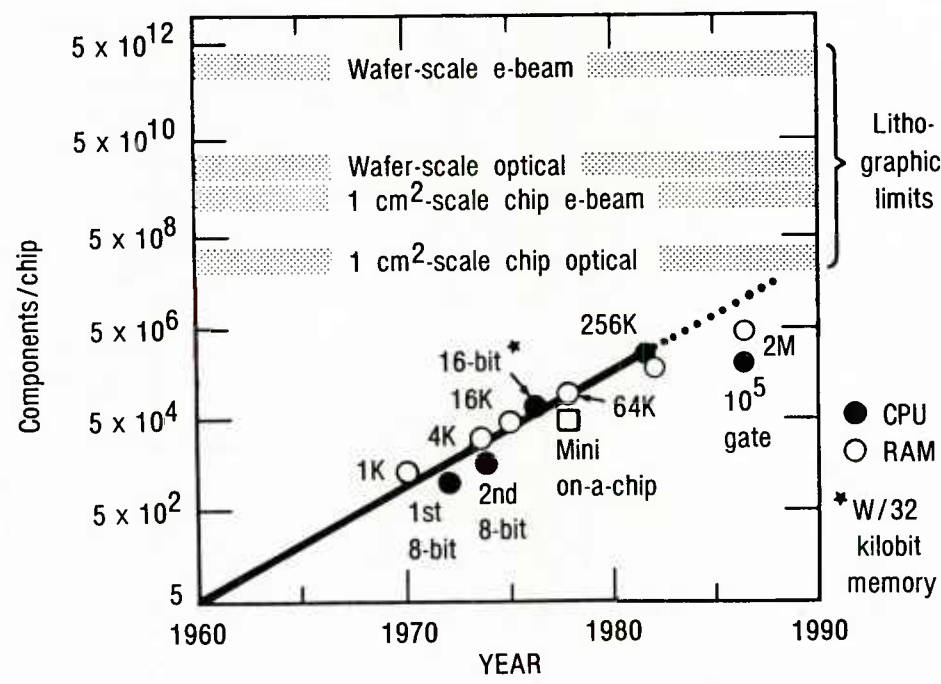
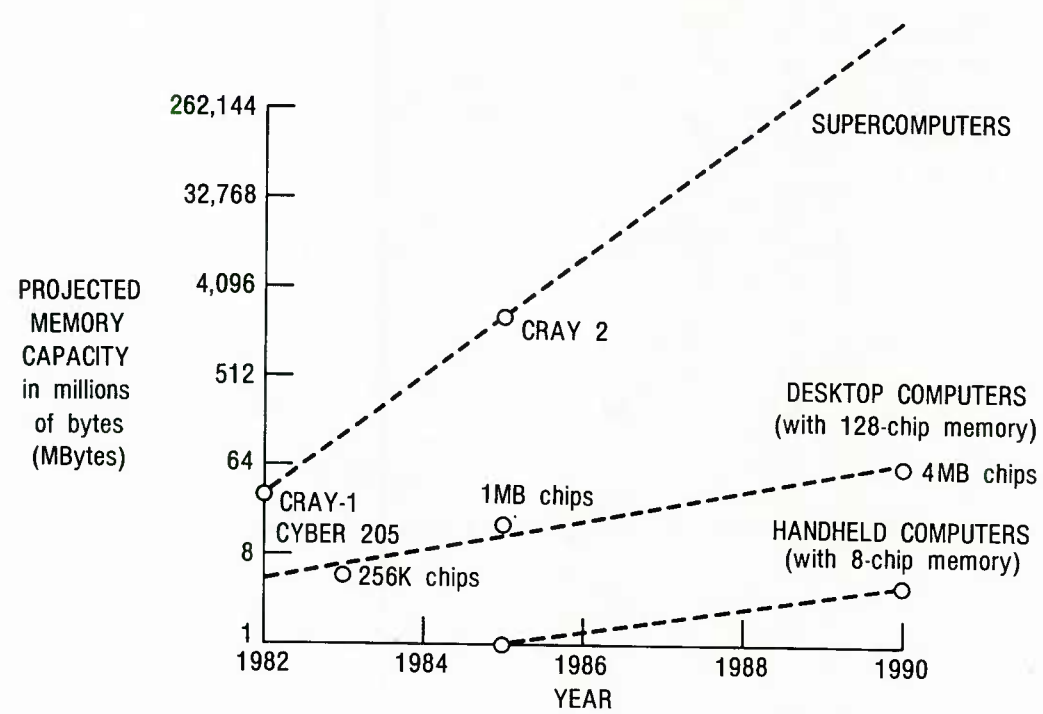
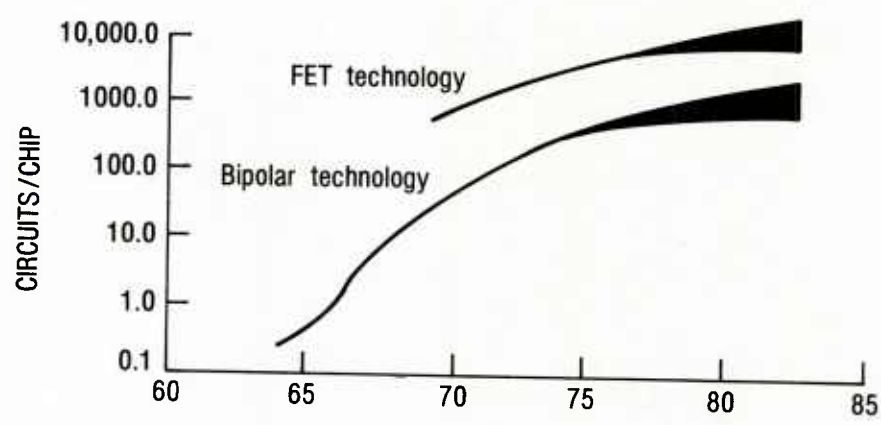
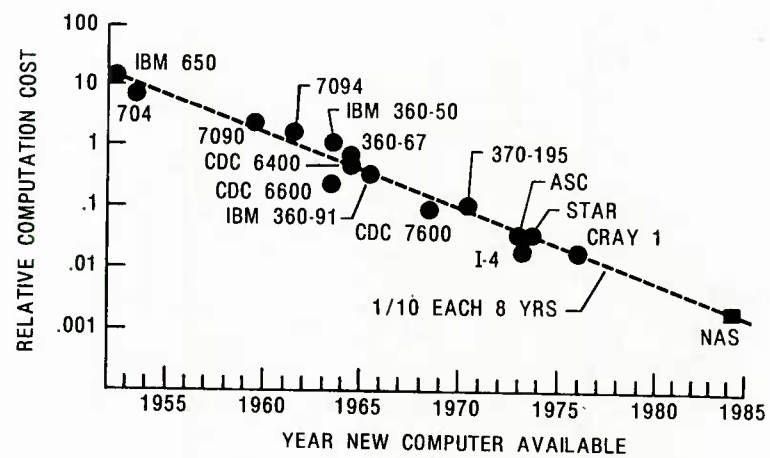


Fig. 2 - Digital logic capability commercially available on a single chip (based on a report by A. C. Haussmann, Lawrence Livermore National Laboratory).



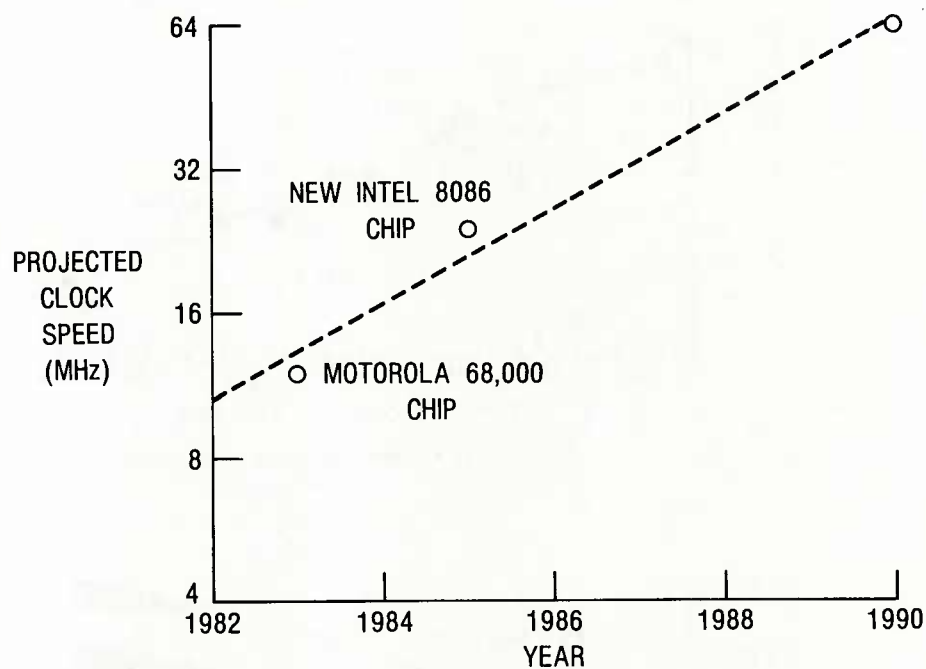


Fig. 6 - Projected clock speed of CPU chips.

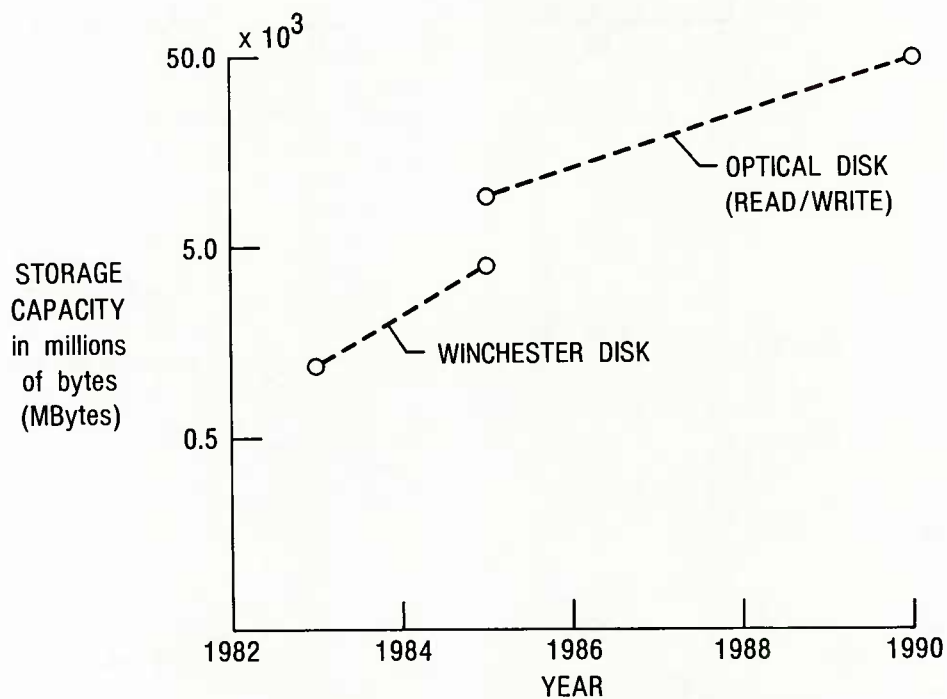


Fig. 7 - Projected disk storage capacity.

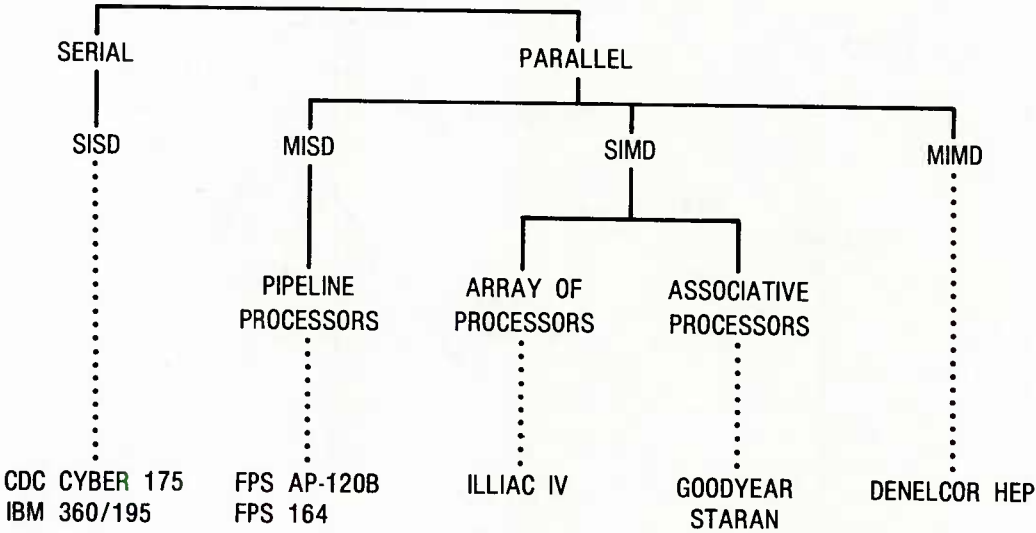


Fig. 8 - Classification of computing systems.

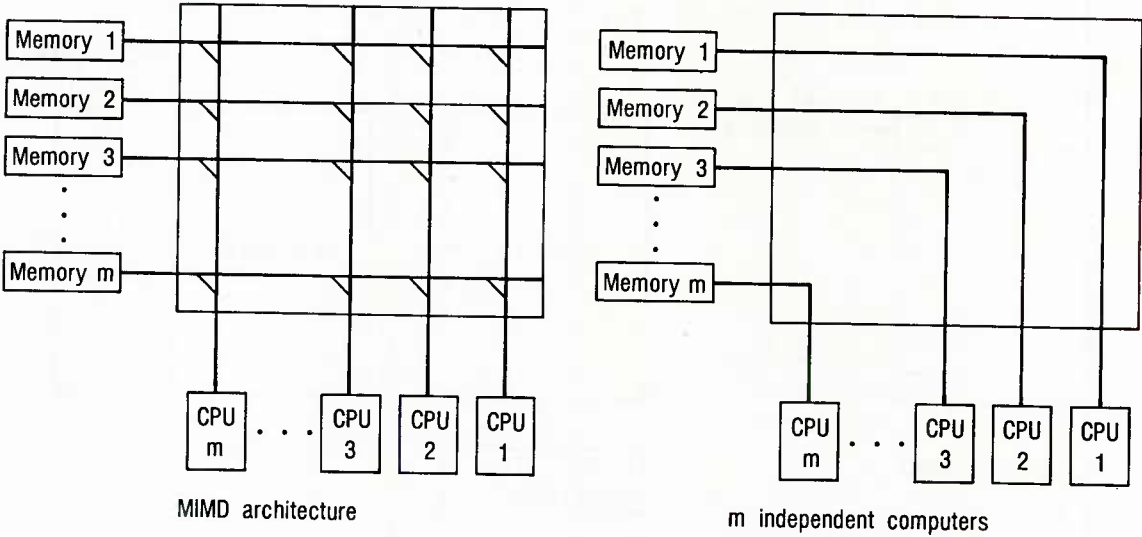
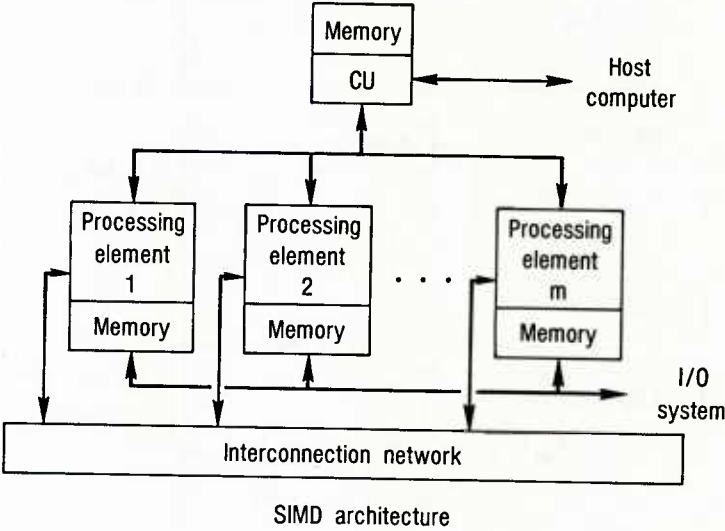


Fig. 9 - Block diagrams for SIMD, MIMD architectures and m independent computers (Ref. 17).

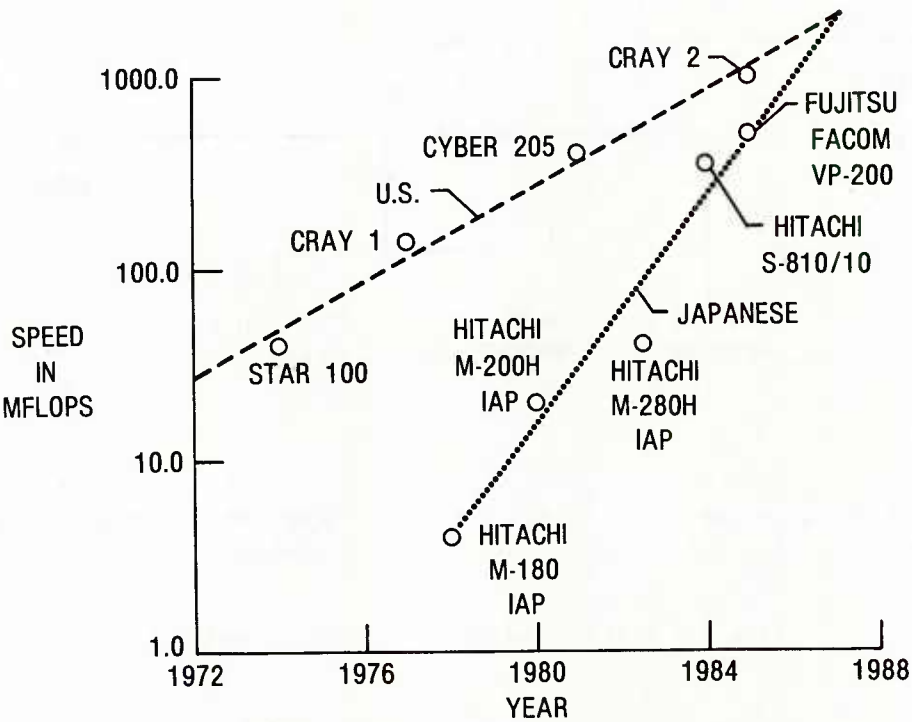


Fig. 10 - Trends in computing speeds for U.S. and Japanese supercomputers.

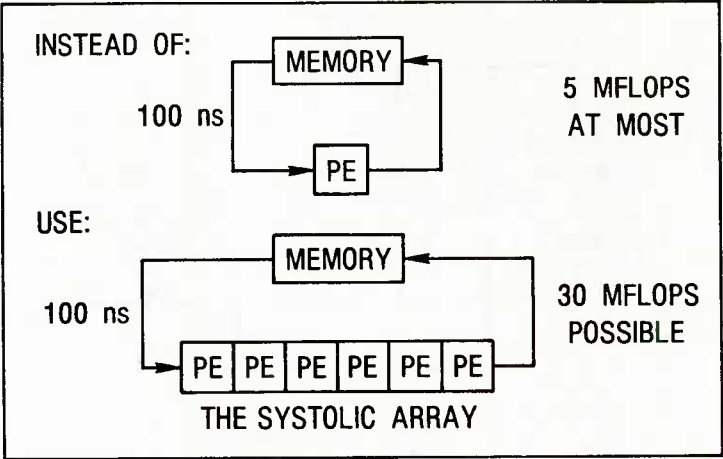


Fig. 11 - Basic principle of a systolic system (Ref. 23).

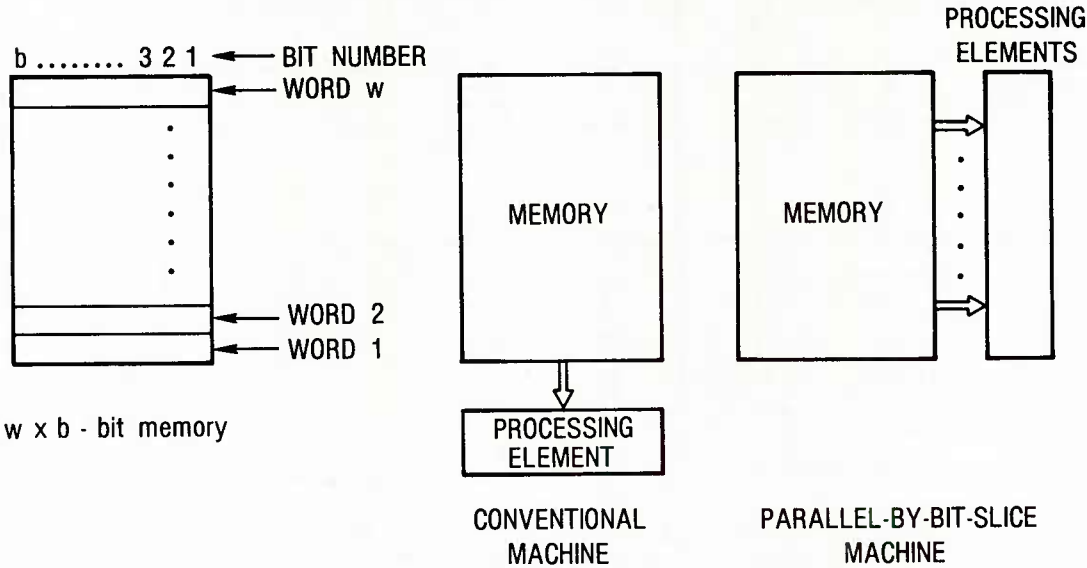


Fig. 12 - Conventional and parallel-by-bit-slice machines.

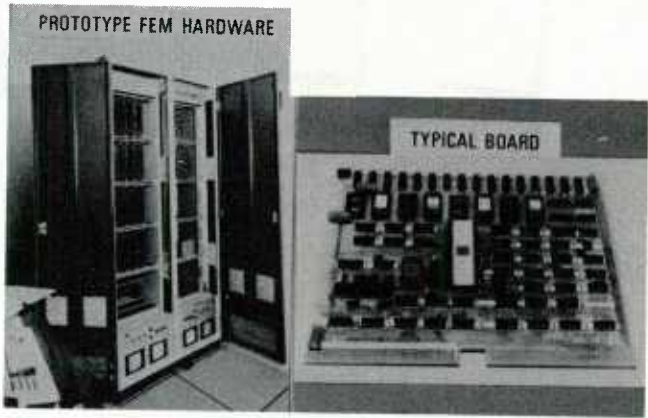


Fig. 13 - Prototype NASA Langley FEM hardware.

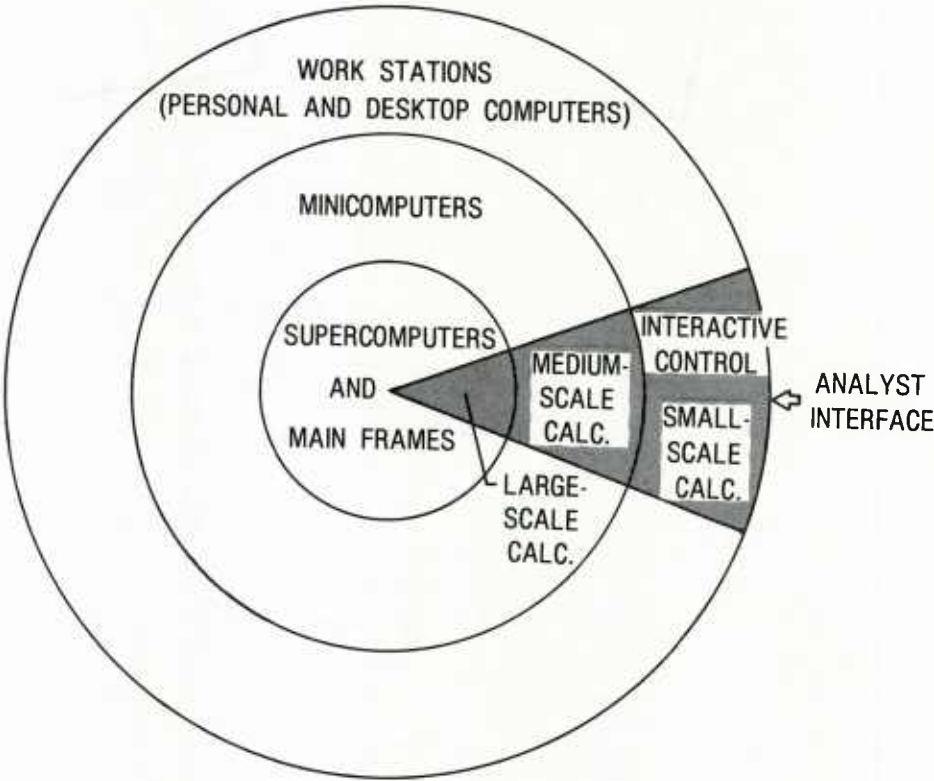


Fig. 14 - Future hardware environment.

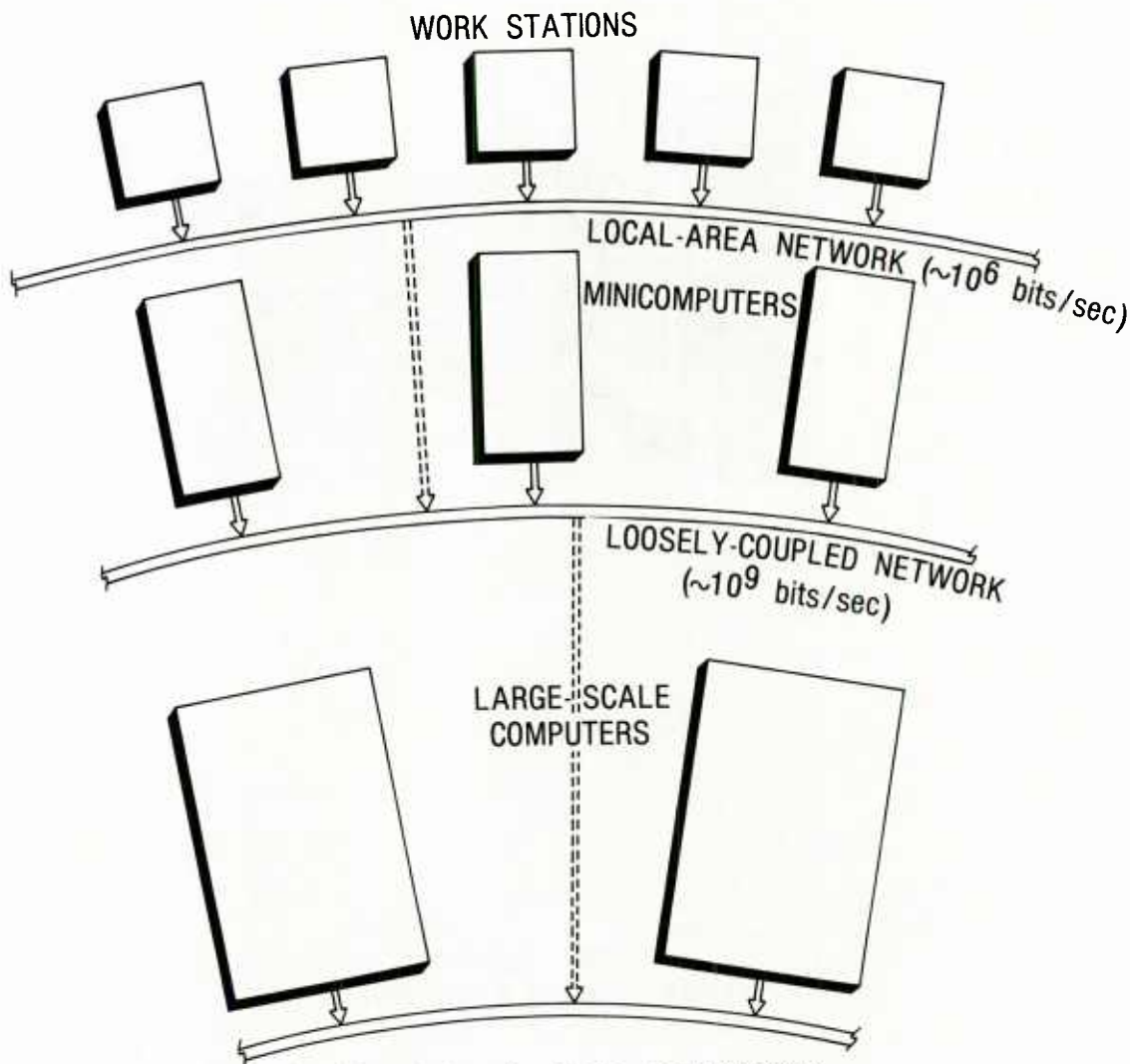


Fig. 15 - Future hardware configuration.

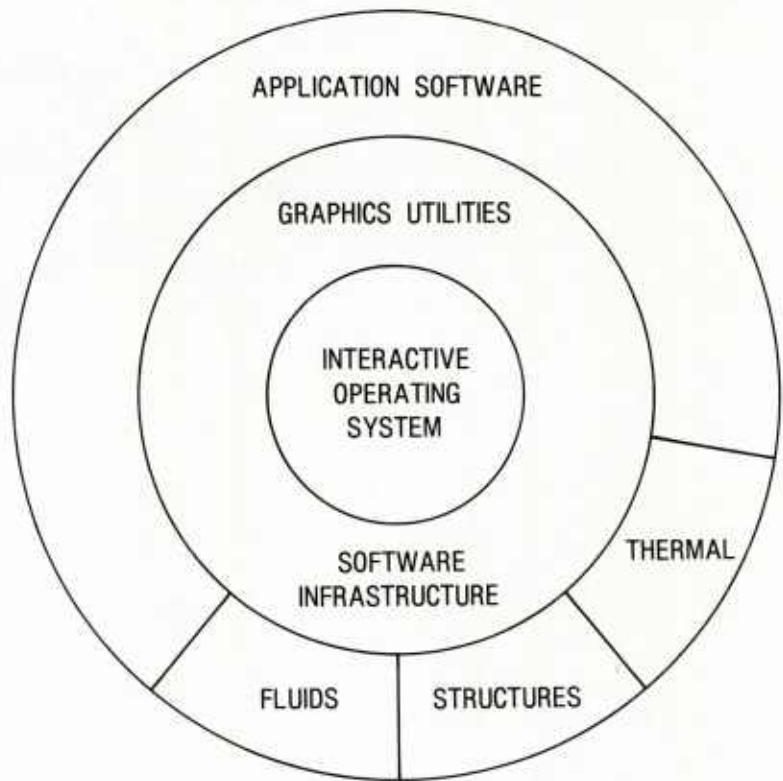


Fig. 16 - Future engineering software system.

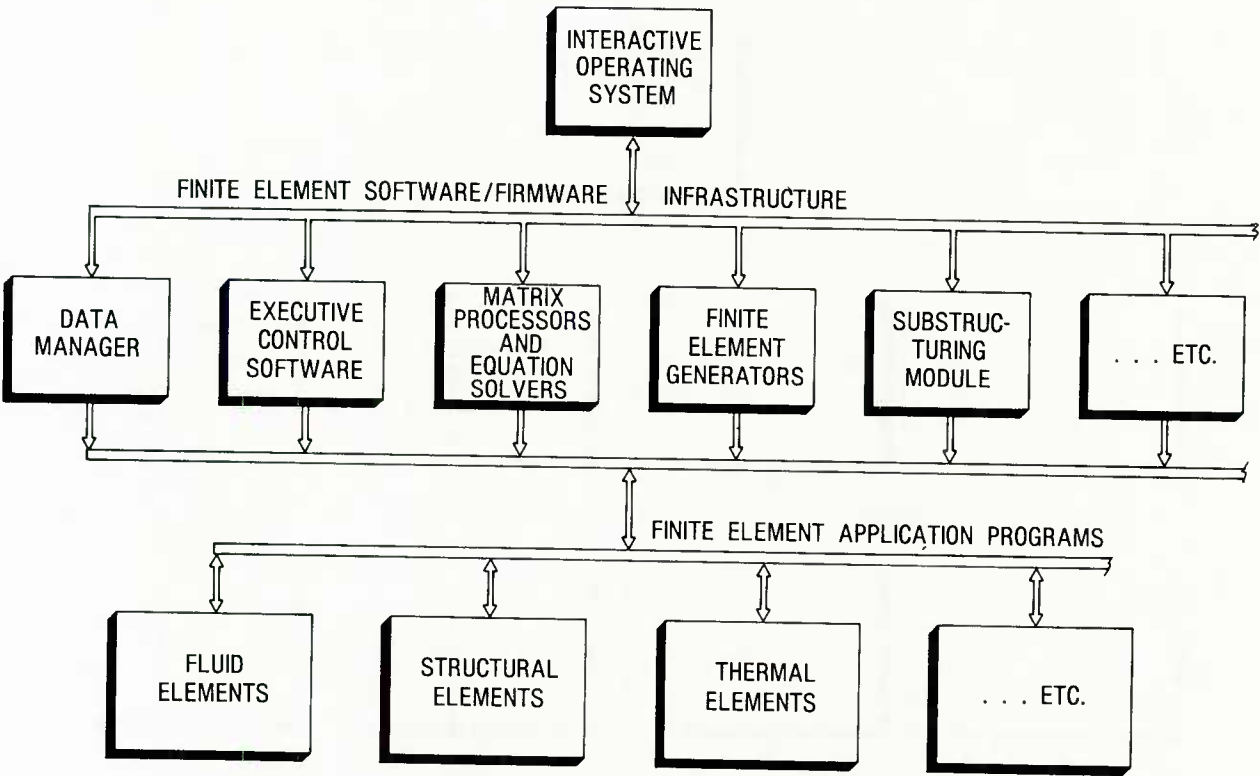


Fig. 17 - Organization of future finite element software/firmware.

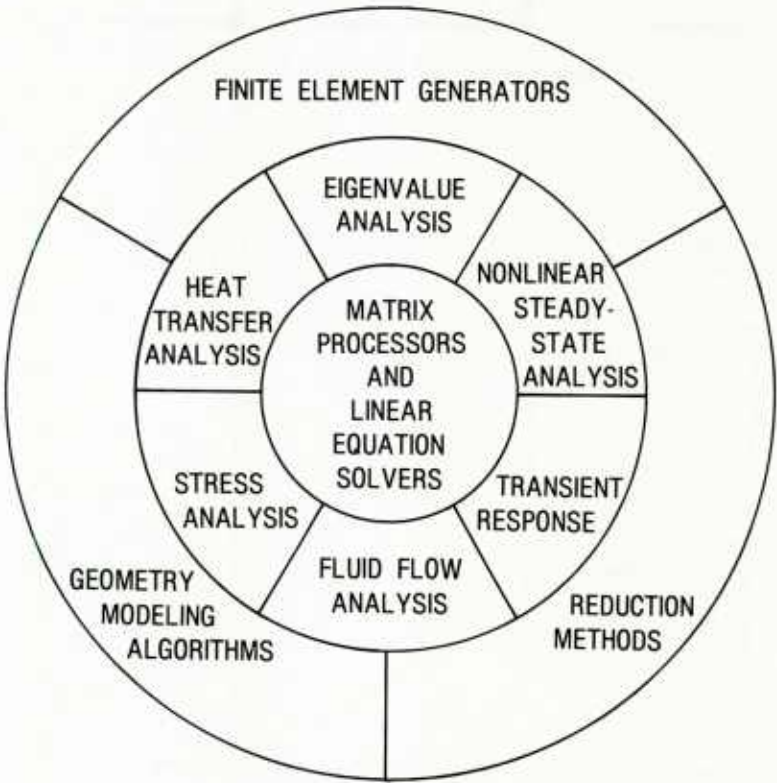


Fig. 18 - Future numerical algorithms.

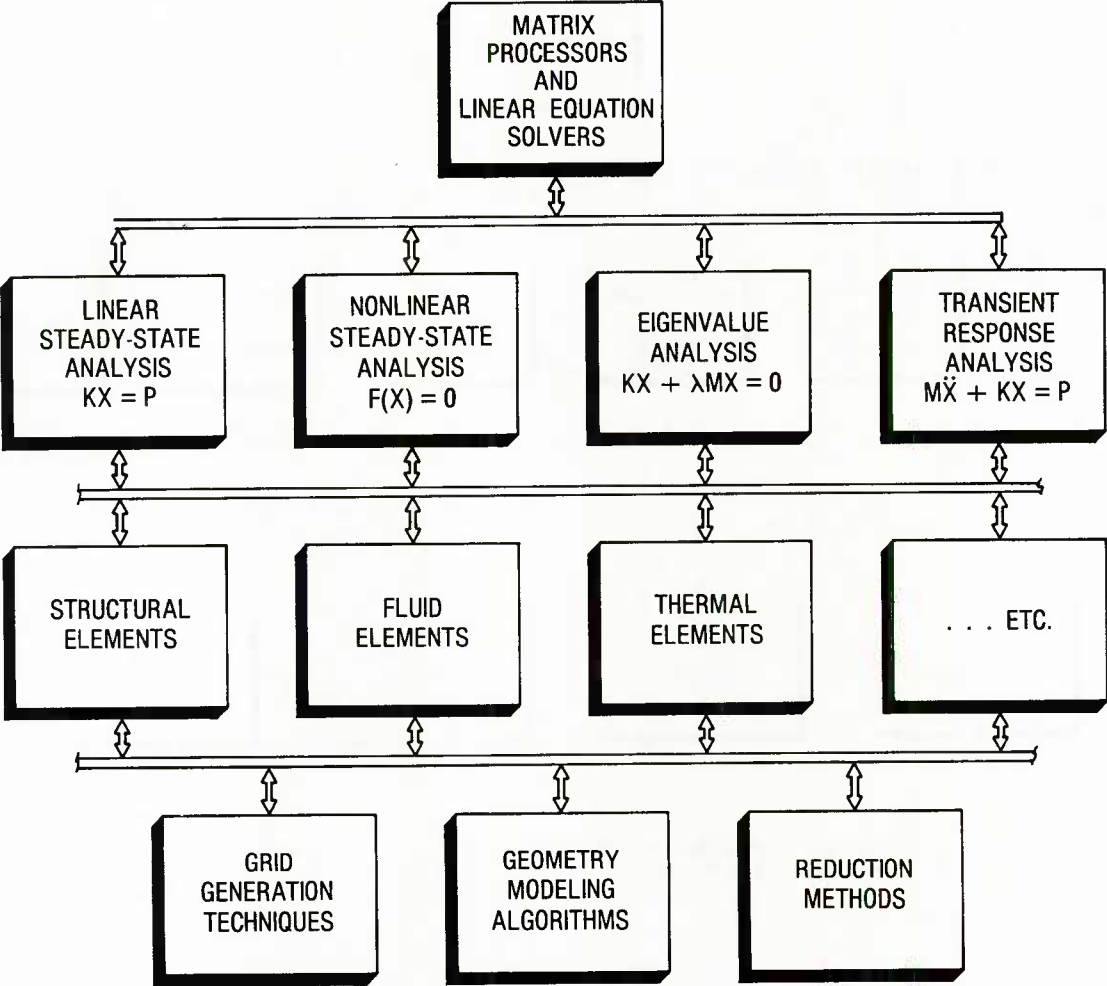


Fig. 19 - Hierarchy of numerical methods and algorithms.

Table 1 - Semiconductor developments (based on copyright information by Integrated Computer Systems - Santa Monica, California)

Date	Number of Transistors	Integration Level	On Chip Device
1960	1	Discrete	Transistor
1964	10's	SSI	Gate
1968	100's	MSI	Gate Arrays
1972	1 000's	LSI	Microprocessors (Calculators)
1976	10 000's	VLSI	Microcomputers
1980	100 000's	ULSI	"Minicomputers"
1984	1 000 000's	ULSI	"Computers"

Table 2 - Architectural features of some of the current and projected supercomputers

	CRAY			CDC CYBER*		Denelcor** HEP1
	1 S	X-MP	2	203	205-444	
Maximum memory capacity (words)	4 M.	4 M.	256 M.	2 M.	4 M.	16 M. program memory 128 M. data memory
Maximum high-speed auxiliary memory capacity (words)	32 M.	32 M.	?	-	-	-
Virtual memory	No	No	No	Yes up to 4×10^{12} words	Yes up to 2×10^{12} words	-
Memory word length (bits)	64	64	64	64/32 (half word)	64/32 (half word)	64
Clock period (ns)	12.5	9.5	4.0	40.0 for vectors 20.0 for scalars	20.0	100/16
Estimated speed (MFLOPS)	150	400	1000	50	400	160 (10 MFLOPS per processor)
Technology chip type	4 chip types	1 chip type	?	40 chip types	29 chip types	
Logic family	SSI/ECL with 1 ns gate delay	SSI/ECL with 750 p.sec. gate delay	SSI/ECL with 650 p.sec. gate delay	LSI/ECL bipolar circuitry for the scalar unit	LSI/ECL bipolar circuitry	SSI/ECL MOS memory
Other features	1 CPU (dedicated for arithmetic operations)	2 CPU's	4 CPU's	2 vector pipelines	1-4 vector pipelines	up to 16 PEMS

*GF-10 of ETA (formerly CDC CYBER 2XX) (1986) will have a peak performance of 10 Gigaflops and uses very high density circuitry and multiple pipelines.

**Denelcor HEP2 (1986) is expected to have a speed of 1 Gigaflop.

Large Scale Computing in Aeronautical Design

by

A.J. Morris
Reader in Aircraft Structures
College of Aeronautics
Cranfield Institute of Technology
Cranfield, Bedford MK43 0AL
United Kingdom

Summary

This publication outlines how use of new advances in both large scale computing and software techniques, can be made to create comprehensive design programs. In particular, the role which can be played by Artificial Intelligence in making large design programs accessible to the single user is emphasised. An AGARD response to the challenge posed by this new situation is discussed and a series of tasks proposed.

1. Introduction

Over the past decade extensive development has taken place in the creation of design oriented computer programs. In the case of structural analysis a variety of major finite element programs have emerged and are used on a routine basis. These systems have been further augmented by the addition of supporting 'user friendly' programs which aid in the data input and results interpretation. The combination of F.E. methods with pre and post processors has lead to the generation of effective design based structural analysis programs. In the case of fluid flow analysis the position is less advanced from the viewpoint of coordinated design programs, but a range of effective independent programs are available. More advanced programs are being developed at the present time which link structural, fluid flow and other programs in automated design or weights estimation programs.

Most of this impressive development has taken place with conventional sequential operation computers. The vector processing computer of the CRAY type offer vastly augmented 'number crunching' capabilities which can extend the capacity of these design systems. Advantage has already been taken of this new capability and a variety of computer programs for the aeroelastic design of aircraft have been converted to employing vector processors. However this capability and the enhanced capability of the next generation of computers naturally prompts the suggestion that a more radical type of design program ought to be developed. This would involve drawing together multi-disciplinary programs within a common database to provide an Integrated Design Program. Such programs have been attempted before but have failed through lack of computing power and lack of suitable program control systems. Use of vector machines alleviates the computer power limitation and the application of new Artificial Intelligence programs offer the possibility of returning control of the programs to the user. The present short paper attempts to outline how such an Integrated Design Program, involving multi-disciplinary algorithms and artificial intelligence, could be brought into being.

The main suggestion involves AGARD taking an initiative and drawing up a costed proposal for an aircraft related I.D.P.. It is envisaged that such a program would be developed and used by organisations in multi-national projects. An appropriate development programme could therefore, be usefully targeted at a consortium approach involving several nations. This could be envisaged as an appropriate procedure for drawing up the initial proposal. In this way groups with special expertise or experience could contribute to the overall task in those areas in which their specific abilities would have the greatest effect.

2. Overview of Current and Potential Advanced Design Capabilities

2.1 Simple applications

The most straightforward and simple applications of vector and array processors is to extend current computer programs to take advantage of the extended power of the 'class 6' machines. Such extensions have taken place in the fields of fluid mechanics and structural analysis.

In the case of computational fluid mechanics impressive results have been achieved in the finite difference solution of flow equations around complex bodies with complex flow regimes. This specific capability with respect to CFM application has often been the driving force behind the acquisition of this type of computer for use in aircraft related research. For structural analysis purposes several of the major finite element packages, including NASTRAN, have been converted to accommodate vector processing. The motivation is to permit these programs to be used for large structural analysis to identify 'hot spots', perform realistic nonlinear analyses and aeroelastic calculations.

This type of structural and CFM application of large computers are highly focussed, endeavouring to solve a specific problem within the overall design cycle. On a slightly wider front structural optimisation optimisation programs are being incorporated into the library of available programs. In this context the vector processing capability can be used to advantage for the repeated structural and aerodynamic analyses required by optimisation routines. Current research is attempting to extend this synthesis role by creating weight estimation and prediction programs within which a structural optimisation program becomes a module in a larger system.

2.2 Integrated Programs

Although much research and development is still required in the creation of the 'simple applications' outlined above it is anticipated that the cost and scope of this work will remain within the resources of a single organisation. The development of more integrated design based programs represents a more expensive option requiring commensurately more extensive investment capabilities. For the purposes of this report the term 'integrated design based programs' implies computer programs which allow a variety of tasks to be performed using information within a common database.

The advantages associated with the use of integrated programs are clear when the interactive nature of the various factors in aircraft design are considered. For example an advanced battlefield helicopter would need good hover endurance, a very high forward speed capability (possibly in excess of 300 knots) and a high 'g' capability. Such a craft would employ advanced blade design with swept tips where the elastic, material and aerodynamic properties of the structure are totally interrelated. In addition a successful overall design would take full account of the interrelation of structural weight, dynamic response, damping factors and other aspects involved in dictating the behaviour of the helicopter fuselage.

Complex problems of this type present the design engineer with both new concepts and increasingly severe constraints. New designs will inevitably involve a range of materials from traditional aluminium alloys, advanced lithium alloys to composite materials including metal/graphite type combinations or composites with yielding resins. The designs could require aeroelastic tailoring capabilities, control on fatigue life and vibrational responses, vulnerability limits, crashworthiness.

The limitations of current software to match such a complex requirement are recognised and efforts are being made by the developers of CAE (CAD/CAM) software to provide programs which will alleviate the situation. This activity has two main thrusts; the multi-disciplinary aspect is being partly covered by generating complex software incorporating several specialist areas, and consideration of the overall design is encouraged through pre-processor software which extends the scope of the applications programs. However this general development in the CAE technology lacks several important features. First the programs emerging are not specific to vector or array processing computers and cannot take full advantage of the speed capability made available by these large machines. Secondly the software is, by its nature, complex and difficult to use to the point where a designer often requires to be an 'expert' in design software to employ the available programs with suitable efficiency. Since this aspect will become worse with further development the design engineer will not be able to acquire the necessary expertise to use the design software. He will therefore require a human interface between himself and the design programs - a highly undesirable state. Thirdly, these systems cannot provide the engineer with the expertise to exploit the new materials and concepts which are introduced into any advanced design. In addition, the long gestation periods associated with the design and development of an aerospace vehicle means that the designer often lacks the experience acquired from involvement in previous designs and studies. We may conclude that the current general development of CAE software is not focussed on large computer technology and will not provide systems which aid the designer to take full advantage of the capabilities provided nor assist him with the difficulties of his design task.

In view of this it is timely for AGARD to consider a new approach to solving design problems through specially created integrated programs which take account of new developments in Artificial Intelligence (A.I.). The integrated programs must be capable of treating the real design problem and the A.I. programs must guide and tutor the user by recourse to Expert Systems.

To facilitate the initial creation of the integrated design software it is felt that some form of 'driving' program should be identified. A logical starting point could involve generating a generalised optimisation system where the objective function could be selected from a variety of parameters including structural weight, performance efficiency, D.O.C., manufacturing costs, etc. This multi-criterion approach would allow the program to be used at a variety of stages in the design cycle. Such a complex requirement will necessitate the creation of a design optimisation program capable of accessing a variety of optimisation and supporting programs. These supporting programs would need to contain modules capable of analysing nonlinear structural behaviour with complex material combinations, be able to describe steady and unsteady aerodynamic flows, have access to manufacturing cost data, etc. All modules would need to be optimised for the performance characteristics of the targeted computer. The resulting program would need extensive graphics facilities and the position of commercially available pre/post processes would need to be examined. It is anticipated that a first stage program would concentrate on CAD and a second would extend the capabilities to the CAM field. The combined capability of the complete program could then be developed to satisfy the requirement that the influence of design changes on manufacturing and production should be assessable.

The development of this type of integrated software will present design engineers with direct access to complex programs related to a range of disciplines outside their own area of expertise and brings us to the second limitation of current CAD systems described above. The effective use of a large scale design program by a single engineer or a small team necessitates that the program itself is able to guide and tutor the user. Producing programs with this capability is a special branch of Artificial Intelligence where effort is devoted to the creation of Expert Systems or Interactive Knowledge Based Systems. These are highly functional programs with a database of knowledge which

is usually represented as a set of facts and rules. In addition the knowledge representation incorporates the logical relationship between the facts and rules which allow inferences and decisions to be made by the system.

The Expert System built-up in this manner from a combination of rules and control programs is intended to stand in place of an expert who solves problems on the basis of knowledge and experience. Clearly the performance of such a system is a function of the quality of derivable knowledge which can be activated. In a realistic knowledge base, derived from human experience, there may be hundreds of interconnecting entries and it is not feasible to solve problems through exhaustive innuementation. A solution can only be achieved by using deductive steps which make connections between the items of knowledge involved in a given phase of the reasoning process. In this way the Expert System mimics the human expert who logically thinks through a problem rather than considering every combination of possibilities which can occur.

A satisfactory solution to the current problem would be achieved by creating an Expert System to guide the user through the complexities of the integrated large scale design program. Because the CAE software being proposed consists of an integrated sequence of programs the Expert System would need to intervene successively to aid the user when each major module is invoked. Such a system would require access to a range of knowledge bases and rules sets pertinent to each of the major parts of the design program. A domain expert would be free to refuse help with his own speciality but could ask for a consultation when the program entered a new domain. It is also envisaged that the individual programs within this complex framework would come under the control of an Expert System operating as a Design Control Program.

Expert Systems can also offer a solution to the third limitation given above concerned with wider questions relating to the designers lack of general design experience or knowledge of a new concept in conjunction with the integrated design program would guide the inexperienced user with knowledge gained from either experienced design engineers or domain experts depending upon the problem being confronted.

The major limitation on the development of this type of major design program is the capacity of the host computer. The 'class 6' machines however, do have the power to allow the construction of usable programs in this area. Further developments in computing capacity will allow for the construction of systems which may be able to deal with the full range of aeronautical problems from initial project study to the programming and running of robotic production lines. At each stage of the programs activity Expert Systems would have an important role to play in producing manageable and effective design programs able to take full advantage of new advances in technology without requiring massive increases in available expert engineers.

3. AGARD Response

The availability of CRAY and similarly powerful computers make the creation of the type of Integrated Design Program, described in section 2, feasible. But an extensive programme of research and development work will be required to define, construct, test and implement a fully integrated large scale system. Because of the costs associated with such a development programme and the scarcity of expertise an effective approach to creating a system could involve a European Consortium with various components of the system being supplied by groups from the members of AGARD. However, the basic viability of the system and the proposed procedure for its construction would need to be established before any commitment could be expected from government or commercial groups.

In view of this it is proposed that AGARD takes an initiative and attempts to define the main constituents of I.D.P. system, establish time scale and qualify costs. The initiative should focus attention on the problems associated with the creation of SDP systems on a European wide basis and attempt to generate new thinking in this area. The final aim should be to produce a costed proposal or similar document which could be used as a basic submission for attracting funding to develop and implement the actual system. An outline of the main tasks associated with this plan are given below.

If AGARD is prepared to take a lead in defining an IDP the above points could constitute a broad set of terms of reference. A suitable channel for carrying out the associated tasks and writing the report would need to be established. Two possible approaches are to either set up a new ad hoc group which can ask individual participants to perform tasks associated with their own speciality, or to tasks associated with their own speciality, or to task a consultant with the entire work plan who would report regularly to the appropriate AGARD Panel. The latter may well be an expensive option for which there is no suitable funding mechanism within AGARD. Alternative approaches to the problem may well be available.

4. Statement of Work

The outline statement given in section 3 can be expanded into a set of main tasks which could be addressed sequentially or in parallel by any group endeavouring to define the IDP system and associated work. These main tasks are given below and represent current thinking on the likely requirements for configuring an IDP system with some Expert System components. However, it should be recognised that the tasks associated with configuring and costing a complex program will change as the problems become more clearly defined.

TASK 1: BASIC LAYOUT OF IDP

The first task requires that the basic layout of the Integrated Design Program should be selected. This includes selecting the 'diver' program which may be an automated design routine and the main modules, i.e. structural analysis, flutter analysis, aerodynamic loads etc. The form of the database would need to be configured and that pre/post processor, graphics routines etc. defined.

TASK 2: SPECIAL SOFTWARE REQUIREMENTS

Because the configuration defined in task 1 is specific to vector or array processor computers the special programming problems associated with optimising the programs for this class of machine will need to be addressed. This includes identifying the areas where particular advantage can be taken of the vector processing capacity. It also includes examining the problems posed by employing the modified languages employed in this class of computer.

TASK 3: ROLE OF COMMERCIALLY AVAILABLE SOFTWARE

Many of the modules and systems which will be defined in task 1 are commercially available i.e. the structural analysis could be performed by NASTRAN. This task is concerned with identifying suitable commercially available programmes, assessing their applicability to the integrated design task and the commercial implications of incorporating them into the IDP. Questions relating to the way in which updates, royalties, etc. would be handled must be examined if the full impact of using existing proprietary systems is to be fully understood.

TASK 4: EXPERT SYSTEMS

This is an extensive task which involves examining the role of Expert Systems within the IDP. Three aspects concerning the use of Expert Systems must be considered:

- (i) Help in using IDP itself.
- (ii) Aid to the design engineer when advanced concepts are employed.
- (iii) Enhancement of the users design experience.

In each case the 'inference engine' and rule base would be presented in outline and the appropriate programming language selected for the tasks and computer.

TASK 5: SYSTEM CONFIGURATION

A complete configuration for the IDP system can now be drawn up on the basis of tasks 1-4. This would involve giving time scales with achievable targets with costs and computing requirements. An important problem associated with large scale design programs is the question of quality assuring the resulting system. In this case the intricate nature of the program, the artificial intelligence aspects and the reliance which will be placed on the system for design requires that effective QA procedures are devised before the program can be used in actual design. The type and mode of use of such procedures would be considered as part of this task.

TASK 6: DETAILED WORK PROGRAMME

Based on task 5 a detailed work programme would be devised matching tasks against available resources. This task would review the consortium approach to creating the system and examine the project management aspects.

TASK 7: REPORT

The final task is to write a costed proposal with a breakdown of packages.

<p>AGARD Report No.706 Advisory Group for Aerospace Research and Development, NATO THE INFLUENCE OF LARGE SCALE COMPUTING ON AIRCRAFT STRUCTURAL DESIGN Published August 1984 54 pages</p> <p>This publication comprises three papers which outline how recent advances in large scale computing capacity could affect the process of aeronautical design.</p> <p>The papers discuss in turn the results of some investigations into the use of vector processing for solving aircraft structural problems, the influence of the new computing systems on computational mechanisms, and the part</p> <p>P.T.O.</p>	<p>AGARD-R-706</p> <p>Aircraft Airframes Structural design Computation Computer programs</p>	<p>AGARD Report No.706 Advisory Group for Aerospace Research and Development, NATO THE INFLUENCE OF LARGE SCALE COMPUTING ON AIRCRAFT STRUCTURAL DESIGN Published August 1984 54 pages</p> <p>This publication comprises three papers which outline how recent advances in large scale computing capacity could affect the process of aeronautical design.</p> <p>The papers discuss in turn the results of some investigations into the use of vector processing for solving aircraft structural problems, the influence of the new computing systems on computational mechanisms, and the part</p> <p>P.T.O.</p>	<p>AGARD-R-706</p> <p>Aircraft Airframes Structural design Computation Computer programs</p>
<p>AGARD Report No.706 Advisory Group for Aerospace Research and Development, NATO THE INFLUENCE OF LARGE SCALE COMPUTING ON AIRCRAFT STRUCTURAL DESIGN Published August 1984 54 pages</p> <p>This publication comprises three papers which outline how recent advances in large scale computing capacity could affect the process of aeronautical design.</p> <p>The papers discuss in turn the results of some investigations into the use of vector processing for solving aircraft structural problems, the influence of the new computing systems on computational mechanisms, and the part</p> <p>P.T.O.</p>	<p>AGARD-R-706</p> <p>Aircraft Airframes Structural design Computation Computer programs</p>	<p>AGARD Report No.706 Advisory Group for Aerospace Research and Development, NATO THE INFLUENCE OF LARGE SCALE COMPUTING ON AIRCRAFT STRUCTURAL DESIGN Published August 1984 54 pages</p> <p>This publication comprises three papers which outline how recent advances in large scale computing capacity could affect the process of aeronautical design.</p> <p>The papers discuss in turn the results of some investigations into the use of vector processing for solving aircraft structural problems, the influence of the new computing systems on computational mechanisms, and the part</p> <p>P.T.O.</p>	<p>AGARD-R-706</p> <p>Aircraft Airframes Structural design Computation Computer programs</p>

<p>which can be played in the design process by Artificial Intelligence software. The third paper goes on to discuss how AGARD might respond to the challenge posed by the present situation.</p> <p>Papers presented at the 58th Meeting of the Structures and Materials Panel, in Sienna, Italy, 2–6 April 1984.</p> <p>ISBN 92-835-0364-3</p>	<p>which can be played in the design process by Artificial Intelligence software. The third paper goes on to discuss how AGARD might respond to the challenge posed by the present situation.</p> <p>Papers presented at the 58th Meeting of the Structures and Materials Panel, in Sienna, Italy, 2–6 April 1984.</p> <p>ISBN 92-835-0364-3</p>
<p>which can be played in the design process by Artificial Intelligence software. The third paper goes on to discuss how AGARD might respond to the challenge posed by the present situation.</p> <p>Papers presented at the 58th Meeting of the Structures and Materials Panel, in Sienna, Italy, 2–6 April 1984.</p> <p>ISBN 92-835-0364-3</p>	<p>which can be played in the design process by Artificial Intelligence software. The third paper goes on to discuss how AGARD might respond to the challenge posed by the present situation.</p> <p>Papers presented at the 58th Meeting of the Structures and Materials Panel, in Sienna, Italy, 2–6 April 1984.</p> <p>ISBN 92-835-0364-3</p>

AGARD

NATO  OTAN7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE
FRANCE

Telephone 745.08.10 · Telex 610176

DISTRIBUTION OF UNCLASSIFIED
AGARD PUBLICATIONS

AGARD does NOT hold stocks of AGARD publications at the above address for general distribution. Initial distribution of AGARD publications is made to AGARD Member Nations through the following National Distribution Centres. Further copies are sometimes available from these Centres, but if not may be purchased in Microfiche or Photocopy form from the Purchase Agencies listed below.

NATIONAL DISTRIBUTION CENTRES

BELGIUM

Coordonnateur AGARD — VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

ITALY

Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
3 Piazzale Adenauer
00144 Roma/EUR

CANADA

Defence Scientific Information Centre

LUXEMBOURG

DE

NASANational Aeronautics and
Space AdministrationWashington, D.C.
20546SPECIAL FOURTH CLASS MAIL
BOOK

Postage and Fees Paid
National Aeronautics and
Space Administration
NASA-451

Official Business
Penalty for Private Use \$300



3 1 43.7, 841012 502276DS
DEPT OF THE NAVY
NAVAL POSTGRADUATE SCHOOL
DUDLEY KNOX LIBRARY
ATTN: SUPERINTENDENT, CODE 1424
MONTEREY CA 93940

FRA

GER

GRE

ICE

Director of Aviation
c/o Flugrad
Reykjavik

Defence Research Information Centre
Station Square House
St Mary Cray
Orpington, Kent BR5 3RE

UNITED STATES

National Aeronautics and Space Administration (NASA)
Langley Field, Virginia 23365
Attn: Report Distribution and Storage Unit

THE UNITED STATES NATIONAL DISTRIBUTION CENTRE (NASA) DOES NOT HOLD
STOCKS OF AGARD PUBLICATIONS, AND APPLICATIONS FOR COPIES SHOULD BE MADE
DIRECT TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS) AT THE ADDRESS BELOW.

PURCHASE AGENCIES*Microfiche or Photocopy*

National Technical
Information Service (NTIS)
5285 Port Royal Road
Springfield
Virginia 22161, USA

Microfiche

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris, France

Microfiche or Photocopy

British Library Lending
Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
England

Requests for microfiche or photocopies of AGARD documents should include the AGARD serial number, title, author or editor, and publication date. Requests to NTIS should include the NASA accession report number. Full bibliographical references and abstracts of AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
published by NASA Scientific and Technical
Information Branch
NASA Headquarters (NIT-40)
Washington D.C. 20546, USA

Government Reports Announcements (GRA)
published by the National Technical
Information Services, Springfield
Virginia 22161, USA



Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

ISBN 92-835-0364-3